Statistical Learning with Applications in Biology

Nielsen, Agnes Martine

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Statistical Learning with Applications in Biology

Agnes Martine Nielsen

Kongens Lyngby 2018
PhD-2018-488

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Richard Petersens Plads, building 324,
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk PhD-2018-488

Summary (English)

Statistical methods are often motivated by real problems. We consider methods inspired by problems in biology and medicine. The thesis is in two parts.

In the first part we consider data in the form of graphs (or networks). These occur naturally in many contexts such as social and biological networks. We specifically consider the setting where we have multiple graphs on the same set of nodes. We propose a model in this setting called the *multiple random dot product graph model*. Fitting the model is an optimization problem which we solve efficiently using a new alternating minimization algorithm. A hypothesis test in the model framework for whether two graphs are drawn from the same distribution is also proposed. Both the fitting algorithm and test are evaluated in simulation studies. The model is also generalized to weighted graphs where we specifically consider Poisson and normally distributed weights. Similar hypothesis tests are proposed in these settings and again we evaluate the performance through simulation studies.

The second part of the thesis considers prediction of disease progression. We compare three common approaches for disease prediction and apply them to a diabetes data set. In this data, the time until a patient goes on to insulin treatment is of interest - especially whether progression is fast or slow. The methods are: A Cox proportional hazards model, a random forest method for survival data, and a neural network approach. The prediction performance, and the pros and cons of the methods are discussed.

Summary (Danish)

Statistiske metoder er ofte motiverede af virkelige problemer. Vi betragter metoder, der er inspirerede af problemer inden for biologi og medicin. Denne afhandling er i to dele.

I den første del betragter vi data i form af grafer (eller netværk). Disse forekommer naturligt i mange kontekster såsom sociale og biologiske netværk. Vi koncentrerer os primært om den situation, hvor vi har flere grafer på det samme sæt knuder. Vi foreslår en model i denne situation kaldet *the multiple random dot product graph*. At fitte denne model er et optimeringsproblem, som vi løser effektivt ved brug af en ny *alternating minimization* algoritme. Inden for denne modelramme forslår vi også en hypotesetest for, om to grafer følger den samme fordeling. Både fittingsalgoritmen og testen er evalueret i simulationsstudier. Modellen er også generaliseret til vægtede grafer, hvor vi især betragter Poisson- og normalfordelte vægte. Vi foreslår lignende hypoteseteste i disse situationer og igen evaluerer vi testen igennem simulationsstudier.

Den anden del af afhandlingen betragter prædiktion af sygdomsprogression. Vi sammenligner tre almindelige tilgange til sygdomsprædiktion og anvender dem på et diabetesdatasæt. I dette data er det tiden, indtil en patient får insulinbehandling, som er interessant - især om progressionen er hurtig eller langsom. Metoderne er: En Cox model, en *random forest* metode for overlevelsesdata og en neurale netværks-strategi. Vi diskuterer prædiktionsevner samt fordele og ulemper af metoderne.

Preface

This thesis was prepared at the Section of Statistics and Data Analysis at the Department of Applied Mathematics and Computer Science at the Technical University of Denmark in partial fulfillment of the requirements for acquiring a PhD degree. The project was supervised by associate professor Line Clemmensen, professor Bjarne Ersbøll, and professor Anders Dahl at DTU Compute.

The thesis deals with statistical methods with applications in biomedical data. The main focus is to develop statistical learning methods and approaches inspired by biological problems and challenges.

The thesis consists of three research papers and an R package documented by its reference manual. For a full list of works and publications associated with the PhD see page vii.

Lyngby, 31-December-2018

Agnes Martine Nielsen

List of Contributions

This thesis is based on three scientific papers and an add-on package for the statistical programming language R (R Core Team, 2015) and a reference manual to support this software package.

[A] **Nielsen, A. M.**, Witten, D. (2018). *The Multiple Random Dot Product Graph Model*, arXiv preprint <arXiv:1811.12172> (Submitted to *Journal of Computational and Graphical Statistics*)

[B] **Nielsen, A. M.**, Svendsen, K. D., Clemmensen, L. (2018) *Hypothesis Testing in the Generalized Multiple Random Dot Product Graph Model*, (draft intended for a statistics journal)

[C] **Nielsen, A. M.**, Nielsen, R. L., Donnelly, L., Zhou, K., Dahl, A. B., Gupta, R., Ersbøll, B. K., Pearson, E., Clemmensen, L. (2018) *A Comparison of Methods for Disease Progression Prediction Through a GoDARTS Study*, (draft intended for a medicine methodology journal)

The **multiRDPG** R package was implemented. It is available at https://cran.r-project.org/package=multiRDPG and the following reference manual accompanies this R package:

[D] **Nielsen, A. M.**, Witten, D. *multiRDPG: Multiple Random Dot Product Graphs*, *R package version 1.0.1*

The following paper has been prepared in collaboration with other researchers during the PhD period.

Nielsen, R., Donnelly, L., **Nielsen, A. M.**, Zhou, K., Tsirigos, K., Ersbøll, B. K., Clemmensen, L., Pearson, E., Gupta, R. (2018), *Prediction of Type 2 Diabetes Progression by Time to Insulin Using Electronic Health Records*, (draft intended for a diabetes journal)

In addition, contributed talks and posters:

- Challenges in Predicting Individual Risk in GWASs, Joint Statistical Meeting 2016, July 30-August 4 2016, Chicago, Illnois, USA (poster)
- Prediction of Disease Progression Using Irregularly Sampled Longitudinal Data, Workshop on Machine Learning for Disease Prediction, May 16-17 2018, Kgs. Lyngby, Denmark (talk)

Acknowledgements

First and foremost, I would like to thank my supervisors: Line Clemmensen for convincing me this was a good idea and for all the help and support; Bjarne Ersbøll for stepping in and having many ideas of possible directions; and Anders Dahl for discussions and advice throughout.

Thank you to Daniela Witten for taking the time to work with me and for guidance and collaboration. Thank you to Ramneek Gupta for countless ideas and discussions. Thank you to Poul V. Andersen Foundation for the PhD collaboration grant received with Rikke Linnemann Nielsen and to Rikke for entering into this collaboration with me. Thank you to Ewan Pearson for working with us. I would also like to thank Exruptive for the collaboration.

Thank you to my colleagues at the Section for Statistics and Data Analysis and in particular to current and former PhD students at the section for many discussions during lunch and coffee breaks. Especially thank you to Sofie P. Jensen for being ready to provide a comment from the neighboring desk from start to (almost) finish. Thank you also to Kira D. Svendsen for stepping in at the neighboring desk and for your keen eye.

Thank you to my partner Gustav Hjerting for the love and support all the way through. And finally thank you to my parents and to all of my family for always being there.

Contents

Chapter 1

Introduction

The field of statistics originated in describing information about demographics of states and hence why the name is related to "state". It developed into describing all kinds of data and information before eventually being extended to also include analysis and method development. It is these last applications of the word "statistics" which constitute the active and expanding research discipline.

Statistical methods are often inspired by practical problems. Around the turn of the 20th century Karl Pearson, frequently regarded as the founder of mathematical statistics, was working on statistical methods focusing on asymptotic theories applicable to data with large sample sizes. During this time, William Sealy Gosset, who studied under Pearson and was employed at the Guinness brewery in Dublin, Ireland, found that these methods were not applicable to his work due to a lack of samples (Boland, 1984). To solve this problem, he developed his own theories and, in order to keep the brewery happy, he agreed to publish them under the enigmatic pseudonym "Student". This gave us, among other things, *Student's t distribution* (Student, 1908). Similarly, we have with a foundation within specific problems in our chosen areas of interest, those of biology and medicine, considered the applicable statistical methods and developed our own when needed.

The thesis consists of two parts. The first part is inspired by a problem that emerges when researchers want to compare multiple networks representing their

data. This could, for example, be a network representing connectivity between brain regions, e.g. Hermundstad et al. (2013). There are several localized regions of the brain and these are connected structurally and functionally. This can be represented as a network (or graph) where the brain regions are the nodes that are connected by edges that describe the connectivity. If we can apply a graph model to data like this, then we can learn the parameters of the model and start comparing different networks. Hermundstad et al. (2013) created brain networks for each of the individual experimental subjects. If the brain contains the same regions for all subjects but the edges are allowed to be different, then we have several graphs on the same set of nodes. It might be the case that we assume all the graphs to be independent and identically distributed but it could also be that we believe the graphs to be different and have related but not identical distributions.

Graphs can be useful for representing data in many different contexts; other examples include social networks and web pages linking to each other. To compare data in the form of graphs we must be able to answer the question of whether they are drawn from the same distribution. So in this thesis we investigate the scientific questions: How can we represent the distributions of multiple graphs on the same set of nodes? And how can we construct a test for the hypothesis that the graphs are drawn from the same distribution? We have developed models for the setting of multiple graphs on the same set of nodes, as well as a hypothesis test. The methods were generalized to also handle weighted graphs because these also often arise.

In the second part of the thesis, we have considered the progression of the treatment of type 2 diabetes patients into the last stage of the treatment where insulin injections are used. Type 2 diabetes is a complex disorder and the progression of it is highly individual. It is therefore important to consider which patient characteristics are predictive of the progression rate, in order to identify what may assist in clinical treatment management. We consider three approaches to disease prediction often seen in medical literature and compare their pros and cons through applying them to the diabetes data set. The scientific question in this study is: How do new machine learning methods for the prediction of disease progression compare to more traditional statistical methods?

This part of the thesis comes from a specific collaboration that I participated in during my PhD studies. Together with PhD student at DTU Bioinformatics, Rikke Linnemann Nielsen, I received the Poul V. Andersen grant for cross-departmental collaboration between PhD Students at DTU. This was awarded to us for a project concerning the application of machine learning methods in disease progression prediction. The collaboration has led to a number of scientific contributions, including two articles. One of these articles is included in this thesis.

1.1 Overview of Thesis

The thesis consists of two parts divided into 8 main chapters as well as three journal articles (A, B and C) and a reference manual for an R-package (D).

Part I consists of the chapters 2 to 5.

Chapter 2 gives a brief introduction to relevant graph concepts and an introduction to random graph models.

Chapter 3 summarizes the contributions in Article A where we introduced a model for multiple random graphs as well as a hypothesis test in that framework. In this chapter, the methods are also illustrated through an example, and the contributions of Article A are discussed.

Chapter 4 gives a detailed description of the use of our R-package `multiRDPG` which implements the methods in Article A.

Chapter 5 summarizes the contributions in Article B where we extend the model from Article A to weighted graphs. In this chapter, the methods are illustrated with an example, and the contributions are discussed.

Part II consists of the chapters 6 and 7.

Chapter 6 gives an introduction to disease prediction and to the approaches to this which are found in literature.

Chapter 7 summarizes the contributions of Article C where common approaches to disease prediction are compared, and the contributions are discussed.

Finally, Chapter 8 wraps up the thesis with a conclusion.

There are 4 appendices attached. Article A is included in Appendix A. The article is entitled "The Multiple Random Dot Product Graph" and is submitted to the Journal of Computational and Graphical Statistics. The article introduces the Multiple Random Dot Product Graphs as an extension of the Random Dot Product Graph and further introduces a hypothesis test for whether two (or more) graphs are drawn from the same distribution within this framework.

Article B is included in Appendix B. The article is entitled "Hypothesis Testing in the Generalized Multiple Random Dot Product Graph Model" and is a draft intended for a statistics journal. The article describes an extension of the Multiple Random Dot Product Graph for weighted networks and considers

two specific examples of Poisson distributed weights and normally distributed weights. It also gives algorithms for a hypothesis test for whether two (or more) graphs are drawn from the same distribution in those two cases.

Article C is included in Appendix C. The article is entitled "A Comparison of Methods for Disease Prediction Through a GoDARTS Study" and is a draft intended for a medicine methodology journal. The article describes a comparison of three common approaches for prediction of disease progression in a diabetes data set. It discusses the limitations and advantages of the three approaches.

Appendix D contains the reference manual for the R-package `multiRDPG`.

Part I

Chapter 2

Random Graphs

The first part of the thesis concerns models for random graphs. In this chapter, we therefore review representations and terminology of graphs in the first section before moving on to models for graphs. We review a selection of random graph models for unweighted and undirected single graphs as well as their extensions to weighted graphs.

2.1 Graphs

A graph (or network), $G(V, E)$, consists of vertices (or nodes), V, and edges, E, between them. The edges may be either directed or undirected resulting in a directed or an undirected graph (Figure 2.1).

In a directed graph the edges have an orientation of going from one node to another. An example of data in the form of a directed graph is a set of webpages linking from one page to the other (or both ways). In this case, the pages are the nodes and there is a clear direction of the link or edge. An undirected graph is simpler than a directed graph as the edges have no orientation. An example of data in the form of an undirected graph is a group of scientific authors who co-author papers. In this case, the authors are the nodes and we have an

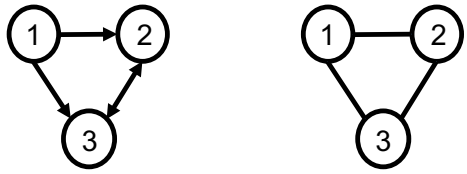


Figure 2.1: Example of a directed graph on the left and an undirected graph on the right.

undirected edge between them if they co-authored a paper. In this thesis, we will only consider undirected graphs.

The edges of graphs can be weighted. This means that if an edge exists, we also associate a weight, W, with them.

A graph can be represented with the lists of nodes, V, edges, E, and possibly weights, W. However, a graph can also be represented by an adjacency matrix, A. If the graph has n nodes and the edges are undirected then the adjacency matrix $A \in \mathbb{R}^{n \times n}$ is an $n \times n$ symmetric matrix. If the graph is unweighted then $A_{ij} \in \{0, 1\}$ such that if there is an edge between the vertices i and j then the matrix element $A_{ij} = A_{ji} = 1$ and if not then $A_{ij} = A_{ji} = 0$. Each element of A is binary and so we call A binary. An example of a graph and its corresponding adjacency matrix can be seen in Figure 2.2. In this figure, we see that node 3 is connected to itself so $A_{33} = 1$. This is called a self-loop. We will allow self-loops.

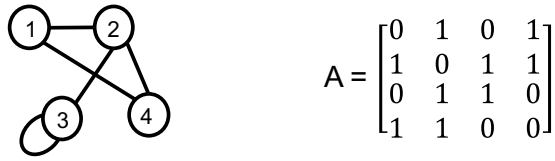
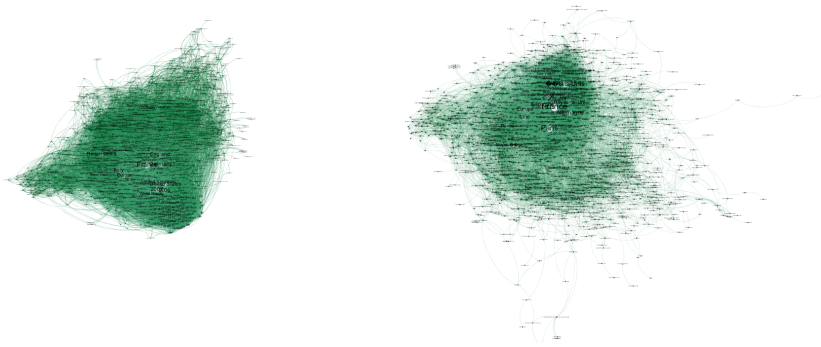


Figure 2.2: The graph sketched on the left results in the adjacency matrix on the left.

If the graph is weighted then if there is an edge between the vertices i and j with weight w_{ij} then $A_{ij} = A_{ji} = w_{ij}$. The adjacency matrix is then no longer binary but still symmetric.

2.1.1 Example: Wikipedia data

As an example we consider two graphs representing subsets of Wikipedia in English and French (Suwan et al., 2016). We will return to this example throughout the first part of the thesis. The data can be found at `http://www.cis.jhu.edu/~parky/Data/data.html`. Each of the vertices represent a page on a topic available in both French and English. The graphs are inherently directed as one page links to the other but here we will consider an undirected and unweighted version of the networks which are constructed such that there is an edge between two pages (nodes) if either links to the other. That means there is one graph representing a subset of the English Wikipedia and another graph representing the French subset. The nodes correspond to each other in the two graphs. The graphs are shown in Figure 2.3 which illustrates the the complexity of the graphs. The nodes are not shown in the same places in Figure 2.3a and Figure 2.3b.



(a) English **(b)** French

Figure 2.3: Wikipedia graphs illustarted using Gephi (Bastian et al., 2009).

We create adjacency matrices for each of the graphs, which we denote A^{English} and A^{French}. As the graphs are unweighted and undirected the adjacency matrices become binary and symmetric. For the English graph, part of the adjacency

matrix is,

$$A^{\text{English}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix},$$

and we similarly display part of the adjacency matrix corresponding to the French graph,

$$A^{\text{French}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Even from just this small subset, we can see that parts of the matrices are equal, e.g $A^{\text{English}}_{2,5} = A^{\text{French}}_{2,5} = 1$, while other parts are different, e.g. $A^{\text{English}}_{5,7} = 1 \neq A^{\text{French}}_{5,7} = 0$.

Table 2.1: Title of the articles associated with the first 7 nodes of the Wikipedia graphs.

Node	English	French
1	Afghanistan	Afghanistan
2	Andorra	Andorre
3	Astronomy	Astronomie
4	Asia	Asie
5	Africa	Afrique
6	Amsterdam	Amsterdam
7	German language	Allemand

In Table 2.1 the title of the articles of the first 7 nodes of the graphs are shown and so we can see that $A^{\text{English}}_{5,7} = 1 \neq A^{\text{French}}_{5,7} = 0$ means that there was a link

between "Africa" and "German language" in the English Wikipedia but not in the French. We return to this example in Section 3.1.2.

2.2 Random Graph Models

After having reviewed the graph terminology, we now consider random graph models. These are of interest because we can use them to describe observed graphs or we can generate random graphs from the models. In the case of unweighted graphs, we can have random graph models which describe the probability of an edge between two nodes. In the literature several models have been proposed.

The Erdös-Rényi graph is the simplest (Erdös and Rényi, 1959). In this model all possible edges are assumed equally likely. That means that

$$P(A_{ij} = 1) = \pi, \tag{2.1}$$

where $\pi \in [0, 1]$. The Erdös-Rényi model is often useful as a null model because it is so simple. However, actual observed graphs are rarely this simple. In many cases, graphs exhibit community structures. Consider for example social networks where local communities will have many connections between the members but not be as closely related to other communities.

The stochastic block model is a simple model for this (Holland et al., 1983). In this model, each node is assigned a class. The probability of an edge between two nodes depends on their class memberships. So let $\tau \in \{1, \ldots, M\}^n$ denote the class membership vector such that each of the n nodes are assigned one of the M classes, and $B \in [0, 1]^{M \times M}$ denote the block connectivity matrix, i.e. a matrix of probabilities for edges between nodes in the classes. The model is,

$$P(A|\tau, B) = \prod_{i<j} B_{\tau_i,\tau_j}^{A_{ij}} (1 - B_{\tau_i,\tau_j})^{1-A_{ij}} \tag{2.2}$$

This means that all edges between nodes within a class are equally likely and that all edges between two given classes are equally likely. By setting all within block and between block edges probabilities equal it reduces to the Erdös-Rényi model.

Another type of model, which further generalizes the stochastic block model is the latent space models (Hoff et al., 2002; Hoff, 2008; Ma and Ma, 2017; Wu et al., 2017). In these models, each node is assigned a position in a latent space and the probability of an edge between two nodes depends on their positions.

The random dot product graph (RDPG) is a latent space model(Young and Scheinerman, 2007; Nickel, 2008; Scheinerman and Tucker, 2010). In RDPG, each node is assigned a vector. The probability of an edge between two nodes is a function of the dot product of their corresponding vectors. That is,

$$P(A_{ij} = 1) = f(x_i^T x_j), \tag{2.3}$$

where $x_1, \ldots, x_n \in R^d$ are d dimensional vectors associated with each of the n nodes and $f : \mathbb{R} \to [0, 1]$ is a link function. This model is of particular interest to us because the model proposed in Article A builds upon this model. We will therefore briefly consider how the RDPG model is fitted. If f is the identity then the following optimization problem has been proposed for fitting the model, i.e. estimating the vectors $x_1, \ldots, x_n$ (Scheinerman and Tucker, 2010),

$$\underset{X \in \mathbb{R}^{n \times d}}{\text{minimize}} \, \|A - XX^T\|_F^2, \tag{2.4}$$

where $\|\cdot\|_F$ is the Frobenius norm and $X \in \mathbb{R}^{n \times d}$ is a matrix whose rows are the vectors $x_1, \ldots, x_n$.

The problem can be solved using the eigenvalue decomposition (Scheinerman and Tucker, 2010). Because A is real and symmetric it can be diagonalized as $A = VDV^T$ where $V \in \mathbb{R}^{n \times n}$ contains the eigenvectors and $D = \text{diag}(\alpha_1, \ldots, \alpha_n)$ is an $n \times n$ diagonal matrix containing the eigenvalues α_i in non-increasing order, i.e $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$. We let $D^+ = \text{diag}(\max(0, \alpha_1), \ldots, \max(0, \alpha_n))$ be an $n \times n$ diagonal matrix containing the positive eigenvalues. We further define $D^+_{[1:d]} = \text{diag}(\max(0, \alpha_1), \ldots, \max(0, \alpha_d))$ as the first d rows and columns of D^+, and $V_{[1:d]}$ as an $n \times d$ matrix which contains the first d columns of V, i.e. the eigenvectors corresponding to the positive eigenvalues in $D^+_{[1:d]}$. In terms of the Frobenius norm, $V_{[1:d]} D^+_{[1:d]} V^T_{[1:d]}$ is the best approximation of A by a positive semi-definite matrix of at most rank d (Scheinerman and Tucker, 2010). The solution to the problem in (2.4) is then $\hat{X} = V_{[1:d]}(D^+_{[1:d]})^{1/2}$.

We now know how to solve it. We will also rewrite the optimization problem in a more convenient form which we will use later. As we are approximating with a positive semi-definite matrix we can equivalently approximate only the positive semi-definite part of A. We can therefore substitute A with $A_+ = VD^+V^T$ in the optimization problem without changing the solution. This gives,

$$\underset{X \in \mathbb{R}^{n \times d}}{\text{minimize}} \, \|A_+ - XX^T\|_F^2. \tag{2.5}$$

As XX^T is positive semi-definite, it can be written in terms of an orthogonal matrix and a diagonal matrix with non-negative diagonal elements. Letting $X = U\Lambda^{1/2}$ where U is an $n \times d$ orthogonal matrix and Λ is a $d \times d$ diagonal

matrix with non-negative elements, then the optimization problem for fitting the RDPG (2.4) can be rewritten and we can equivalently solve the problem,

$$\underset{U\in\mathbb{R}^{n\times d},\, U^TU=I,\, \Lambda\in\Delta_+^d}{\text{minimize}} \|A_+ - U\Lambda U^T\|_F^2, \tag{2.6}$$

where Δ_+^d is the set of diagonal $d \times d$ matrices with nonnegative diagonal elements. The solution to (2.4) is then $\hat{X} = \hat{U}\hat{\Lambda}^{1/2} = V_{[1:d]}(D^+_{[1:d]})^{1/2}$. We will return to this version of the problem in Chapter 3.

2.3 Weighted Random Graph Models

In the case of weighted graphs, we are not modeling the probability of an edge but the weights of the edges. Here we will consider the weighted extensions of the random graph models discussed in Section 2.2.

The Erdös-Rényi model (Erdös and Rényi, 1959) has that all graphs with the same number of edges are equally probable. Garlaschelli (2009) extends this model to weighted graphs by creating a model where the graphs with the same total sum of weights are equally probable. They let the probability that any two nodes are joined with an edge with weight w be

$$q(w) = p^w(1-p), \tag{2.7}$$

meaning that the edge weights follow a geometric distribution with parameter $1-p$.

The stochastic block model has been extended to weighted graphs by Aicher et al. (2013). In the weighted stochastic block model they let the parameters of the distribution of the edge weights be dependent on the class memberships. Let $\tau \in \{1,\dots,M\}^n$ denote the class memberships, $\mathcal{R}$ be a partioning of the edges into R disjoint sets where there is one set for each pair of blocks, let $\theta_r \in \theta = \{\theta_1,\dots,\theta_R\}$ be the parameters of the distribution of the edge weights in the rth set in $\mathcal{R}$, and let q be a probability distribution. The model then is,

$$P(A|\tau,\theta,\mathcal{R}) = \prod_{i<j} q(A_{ij}|\theta_{\mathcal{R}(\tau_i,\tau_j)}) \tag{2.8}$$

Aicher et al. (2013) restrict the distributions to be from the exponential family for simplicity but that does include many relevant distributions, e.g. normal and Poisson distributions.

The random dot product graph has also been extended to unweighted graphs (DeFord and Rockmore, 2016; Tang, 2017). This has been done by

letting the weights follow a probability distribution and modeling each parameter of the distribution with a dot product. Let $\text{dist}(p^1, \ldots, p^L)$ be a parametric probability distribution with parameters p^l for $l = 1, \ldots, L$, and let d^l specify the number of latent dimensions for the lth parameter. The model then is,

$$A_{ij} \sim \text{dist}(p^1, \ldots, p^L), \quad p^l = (x_i^l)^T x_j^l \tag{2.9}$$

where $A_{ji} = A_{ij}$, $A_{jj} = 0$, and $x_1^l, \ldots, x_n^l \in \mathbb{R}^d$ for $l = 1, \ldots, L$ are d-dimensional vectors associated with the n nodes. If the distribution dist is chosen to be the Bernoulli distribution then the model reduces to the RDPG (2.4). In this model any probability distribution can be used but estimation of the parameters can be difficult. When estimating the parameters DeFord and Rockmore (2016) only consider the distribution to be the Poisson distribution for simplicity.

Chapter 3

The Multiple Random Dot Product Graph

In the previous chapter, we considered random graph models for single graphs. In this chapter, we will consider the random graph model developed in Article A which is a model for multiple unweighted and undirected graphs. It was inspired by many biological (and other) scenarios where we have two or more networks that we think are closely related and whose distributions we want to compare.

First, the model is described as well as how it can be fitted; after that we move on to a hypothesis test in this framework; and finally, we discuss the contributions.

3.1 The Multiple Random Dot Product Graph

The graph models described in Section 2.2 model a single unweighted and undirected graph or multiple graphs that are assumed to be independent and identically distributed. The model we will consider here models multiple unweighted and undirected graphs on a common set of vertices. This is a setting that appears often in research and for which several models have been proposed (Tang et al., 2009; Shiga and Mamitsuka, 2012; Dong et al., 2014; Durante et al., 2017; Wang et al., 2017). One of those is the Multiple Random Eigen Graph (MREG)

model (Wang et al., 2017),

$$P\left(A_{ij}^k = 1\right) = f\left(W_{ij}^k\right), \quad W^k = U\Lambda^k U^T, \quad k = 1, \ldots, K, \tag{3.1}$$

where $A^1, \ldots, A^K \in \{0,1\}^{n \times n}$ denote K adjacency matrices on a single set of n nodes, U is a $n \times d$ matrix in which the ith row is the d-dimensional vector corresponding to the ith node across all K graphs, $\Lambda^1, \ldots, \Lambda^K$ are $d \times d$ diagonal matrices, and $f : \mathbb{R} \to [0,1]$ is a link function. This is an extension of the RDPG because if $\Lambda^1 = \ldots = \Lambda^K$ and if $W^1 = \ldots = W^K$ are positive semi-definite then it reduces to the RDPG. However, MREG does not generally guarantee that $W^1, \ldots, W^K$ are positive semi-definite because it does not restrict the diagonal elements of Λ^k to be positive and it therefore does not generally reduce to RDPG for $K = 1$. In Article A, we therefore proposed the multiple random dot product graph (multi-RDPG) model, which guarantees this. The multi-RDPG model takes the form

$$P\left(A_{ij}^k = 1\right) = f\left(W_{ij}^k\right), \qquad W^k = U\Lambda^k U^T, \qquad k = 1, \ldots, K, \tag{3.2}$$

where U is an $n \times d$ orthogonal matrix ($n \gg d$), $\Lambda^1, \ldots, \Lambda^K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements, and $f : \mathbb{R} \to [0,1]$ is a link function. We will only consider the link function to be the identity. When $K = 1$ it reduces to the RDPG and is therefore a more direct extension of RDPG than MREG. This also has the advantage that we can interpret the kth graph as lying in a d-dimensional space.

To fit the multi-RDPG model we directly extended the re-written optimization problem (2.6) for fitting RDPG to

$$\underset{U \in \mathbb{R}^{n \times d},\ U^T U = I,\ \Lambda^1, \ldots, \Lambda^K \in \Delta_+^d}{\text{minimize}} \sum_{k=1}^{K} \|A_+^k - U\Lambda^k U^T\|_F^2. \tag{3.3}$$

where Δ_+^d is the set of $d \times d$ diagonal matrices with nonnegative diagonal elements. So for our optimization problem we have a sum because we have K adjacency matrices instead of one.

In Article A we proposed an algorithm for solving this problem (re-stated in Algorithm 1). We showed that every step of the algorithm decreases the objective function in (3.3). Through simulation studies we were able to demonstrate that the algorithm outperforms existing methods for estimating the parameters including the method by Wang et al. (2017) who introduced MREG.

Algorithm 1 Alternating Minimization Algorithm for Solving (3.3) from Article A

1: For $k = 1, \ldots, K$, initialize Λ^k to be a $d \times d$ diagonal matrix with nonnegative diagonal elements.
2: Initialize U^{old} to be an orthogonal $n \times d$ matrix.
3: For $k = 1, \ldots, K$, let VDV^T denote the eigendecomposition of A^k, and define $A^k_+ \equiv VD_+V^T$, where D_+ is the diagonal matrix with diagonal elements $(D_+)_{ii} = \max(D_{ii}, 0)$ for $i = 1, \ldots, n$.
4: **while** not converged **do**
5: Define the matrices B and C to have as their columns the left and right singular vectors, respectively, of the matrix $\sum_{k=1}^{K} A^k_+ U^{\text{old}} \Lambda^k$. Then, update $U \leftarrow BC^T$.
6: For $k = 1, \ldots, K$ and $j = 1, \ldots, n$, update $\Lambda^k_{jj} \leftarrow \max(0, Z_{jj})$, where Z_{jj} is the jth diagonal element of the matrix $Z = U^T A^k_+ U$.
7: Update $U^{\text{old}} \leftarrow U$.
8: **end while**

3.1.1 Computation Times

In order to briefly consider the computation times of Algorithm 1, a small simulation study is set up. In the simulation study in Article A we consider different values of the number of nodes n. So for $n \in \{10, 20, 50, 100, 200\}$ data is generated from (3.2) using

$$U = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & 1 & \ldots & 1 \\ 1 & -1 & 1 & -1 & \ldots & -1 \\ 1 & 1 & -1 & -1 & \ldots & -1 \end{bmatrix}^T, \tag{3.4}$$

Λ^k with $\Lambda^k_{11} \sim \text{Uniform}(0.4n, 0.75n)$, $\Lambda^k_{22} \sim \text{Uniform}(0.05n, 0.2n)$ and $\Lambda^k_{33} \sim \text{Uniform}(0, 0.05n)$, $K = 2$, and f as the identity. This choice of parameters results in $U\Lambda^k U \in [0, 1]^{n \times n}$. The model is then fitted using Algorithm 1 and the algorithm by Wang et al. (2017). Besides these, $\text{RDPG}_{\text{seperate}}$ was obtained by fitting separate RDPG models to each adjacency matrix and RDPG_{all} was obtained by fitting a single RDPG to both adjacency matrices.

Each of these methods was timed and the timings are shown in Figure 3.1. Each fit is repeated 100 times to obtain standard errors. We see that RDPG_{all} and $\text{RDPG}_{\text{seperate}}$ are the fastest which is to be expected as they are computed by finding the eigenvalues and do not require an iterative algorithm. For small n Algorithm 1 and the algorithm by Wang et al. (2017) have similar timings but for larger n Algorithm 1 is faster. We note that, because the Λ^ks are close to equal, the algorithm will be faster than if they were very different.

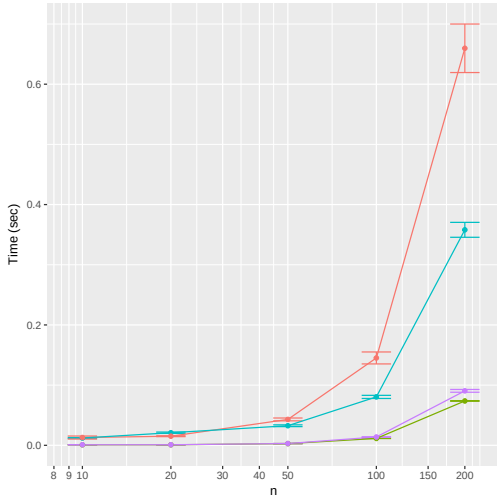


Figure 3.1: Timings of multi-RDPG (Algorithm 1) (blue), MREG (Wang et al., 2017) (red), RDPG_{all} (green), and $\text{RDPG}_{\text{separate}}$ (purple). The figures display the mean and standard error, over 100 simulated data sets of the timings in seconds.

3.1.2 Example: Wikipedia Data

We return to the Wikipedia example from Section 2.1.1. We now want to fit the multi-RDPG model (3.2) to this data set. Fitting the model using Algorithm 1 with the latent dimension $d = 5$ gives us the following parameters besides the shared matrix U,

$$\hat{\Lambda}^{\text{English}} = \begin{bmatrix} 74.7 & 0 & 0 & 0 & 0 \\ 0 & 39.4 & 0 & 0 & 0 \\ 0 & 0 & 25.5 & 0 & 0 \\ 0 & 0 & 0 & 22.7 & 0 \\ 0 & 0 & 0 & 0 & 21.7 \end{bmatrix},$$

$$\hat{\Lambda}^{\text{French}} = \begin{bmatrix} 70.6 & 0 & 0 & 0 & 0 \\ 0 & 23.9 & 0 & 0 & 0 \\ 0 & 0 & 21.1 & 0 & 0 \\ 0 & 0 & 0 & 18.2 & 0 \\ 0 & 0 & 0 & 0 & 14.6 \end{bmatrix}.$$

We notice that all the values for the English graph are larger than those from the French. This may just be due to the English articles being longer rather than the distribution being completely different. It may just be scaled. So because

the English graph has more edges than the French, we try to downsample the English graph. More precisely the English graph has 18,857 edges and the French has 14,973. We randomly downsampled the English graph so that it contains the same number of edges. We then again chose the latent dimension to be $d = 5$. The model is fitted using Algorithm 1. This gives us the following parameters for each of the graphs besides the shared matrix $\hat{U}$,

$$\hat{\Lambda}^{\text{English,sampled}} = \begin{bmatrix} 58.7 & 0 & 0 & 0 & 0 \\ 0 & 30.9 & 0 & 0 & 0 \\ 0 & 0 & 20.4 & 0 & 0 \\ 0 & 0 & 0 & 17.2 & 0 \\ 0 & 0 & 0 & 0 & 18.0 \end{bmatrix},$$

$$\hat{\Lambda}^{\text{French}} = \begin{bmatrix} 71.4 & 0 & 0 & 0 & 0 \\ 0 & 25.4 & 0 & 0 & 0 \\ 0 & 0 & 21.5 & 0 & 0 \\ 0 & 0 & 0 & 19.0 & 0 \\ 0 & 0 & 0 & 0 & 14.9 \end{bmatrix}.$$

We see that now the values of the English are sometimes lower than those of the French and vice versa. We also notice that the values for the French graph changed slightly which is because the English graph and therefore the common U have changed.

We might be interested in quantifying whether the two graphs are drawn from the same distribution, i.e. whether the two Λ's can be said to be equal. Simply looking at the values, we might expect they are not but we cannot answer whether values are close without knowing their distributions.

3.2 Hypothesis Testing for Graphs

There are many examples where we might want to compare whether two (or more) networks are drawn from the same distribution. For example, consider two networks representing two brains, one which has a disease and one which is healthy. We might want to know if they share a distribution. Another example is whether the link networks of Wikipedia are the same in English and French.

In order to test whether two graphs are drawn from the same distribution, we need to set up a hypothesis, decide on a test statistic, and derive a distribution of the test statistic under the null hypothesis. Within the multi-RDPG model the hypothesis becomes,

$$H_0 : \Lambda^1 = \cdots = \Lambda^K \tag{3.5}$$

because if this is true then the graphs are all drawn from the same RDPG which means that the probability of an edge between two nodes is the same across graphs. The test is derived in Article A.

The test statistic was inspired by likelihood ratio tests. It is the minimum of the objective function in (3.3) when $\Lambda^1, \ldots, \Lambda^K$ are all assumed equal minus the minimum when they are not. That is (re-stated here from Article A),

$$T\left(A_+^1, \ldots, A_+^K\right) = \left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda \in \Delta_+^d} \sum_{k=1}^{K} ||A_+^k - U \Lambda U^T||_F^2 \right) - \left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda^1, \ldots, \Lambda^K \in \Delta_+^d} \sum_{k=1}^{K} ||A_+^k - U \Lambda^k U^T||_F^2 \right). \tag{3.6}$$

The final thing needed is a distribution of the test statistic under the null hypothesis. As we do not know the theoretical distribution, we used permutation testing. The idea in permutation testing is to approximate the null distribution by permuting the data to get data under the null hypothesis and then calculate the test statistic for the permuted data. This is repeated to obtain a distribution. Alternatively, one can obtain parametric bootstrap data under the null hypothesis by fitting a model under the null hypothesis and sampling from this model. We have used the latter approach for the model in Chapter 5.

The test for the hypothesis $H_0 : \Lambda^1 = \cdots = \Lambda^K$ is summarized in Algorithm 2 which is restated here from Article A. In that article, we performed simulation studies that showed that the resulting p-values are uniformly distributed under the null hypothesis which suggests that we have adequate type I error control. We also showed that the fraction of times we reject the hypothesis increases when the graphs are drawn from models with increasingly different Λs as well as when the number of nodes in the graphs increase. This indicates that the test has adequate power under the alternative.

We note that as Algorithm 2 is a permutation test it requires Algorithm 1 to be run for each permuted data set. In Section 3.1.1 we saw that the model is fast to fit but testing using Algorithm 2 with $B = 1000$ requires the model to be fit 1000 times which according to Figure 3.1 for $n = 10$ would take approximately 12 seconds and for $n = 200$ approximately 6 minutes. From these numbers we can see that the simulation studies performed in Article A are computationally intensive. In order to show that the p-values were uniformly distributed the test was run 1000 times for each value of n and sets of parameters Λ and U which means that the simulation would take approximately 3.4 hours for $n = 10$ and 9 hours for $n = 50$. For the power calculation these had to be further repeated which is why these simulations were parallelized.

Algorithm 2 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (3.2) from Article A

1. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (3.6).
2. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{1,*b}\right)_{ij}, \ldots, \left(A^{K,*b}\right)_{ij}$ be a random permutation of $\left(A^1\right)_{ij}, \ldots, \left(A^K\right)_{ij}$.
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (3.6).
3. Compute the p-value,

$$p = \frac{1}{B} \sum_{b=1}^{B} I_{\left\{T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right) \geq T\left(A_+^1, \ldots, A_+^K\right)\right\}},$$

where I_C is an indicator function that equals one if the event C holds, and equals zero otherwise.

3.2.1 Example: Wikipedia Data

We again return to the Wikipedia example (see Sections 2.1.1 and 3.1.2). In Article A, we calculated a p-value for the hypothesis that $H_0 : \Lambda^{\text{English,sampled}} = \Lambda^{\text{French}}$ in the setup in this example when we had downsampled the edges of the English graph. We found that $T(A_+^{\text{English,sampled}}, A_+^{\text{French}}) = 93.08$ and a p-value of $p = 0$. We therefore rejected the null hypothesis that the two graphs are drawn from the same distribution. Doing the same with the full English graph gives a test statistic of $T(A_+^{\text{English}}, A_+^{\text{French}}) = 146.53$ and a p-value of $p = 0$. So in that case we unsurprisingly also reject the hypothesis.

In Article A, we also did a comparison of two edge subsets of the English graph. We sampled 15,000 edges twice leading to two largely overlapping graphs. Testing the hypothesis that these two were equal gave a test statistic of $T(A_+^{\text{sample 1}}, A_+^{\text{sample 2}}) = 0.08$ and a p-value of $p = 0.81$. As expected we fail to reject the null hypothesis.

3.3 Discussion

The multi-RDPG model presented in this chapter is a model for multiple random graphs on the same set of nodes. In the model, the graphs are closely related because we assume that they have the same latent vectors in U but they are allowed to have different diagonal values in Λ^k. This model is restrictive compared to fitting separate models to each graph but appropriate if we believe that the graphs are related. In the optimization problem (3.3) and in Algorithm 1 the link function f is assumed to be the identity. It does therefore not necessarily map to $[0, 1]$. We chose this function for simplicity. The same choice has been made previously for the RDPG (Scheinerman and Tucker, 2010). It is, however, a more natural choice to choose a function that maps into $[0, 1]$ such as the logistic function, $f(x) = \exp(x)/(1 + \exp(x))$. This choice would change the optimization problem and thereby also the fitting algorithm. This link function has been used in RDPG (O'Connor et al., 2017) and it is potential future work to do the same for multi-RDPG. Another direction which this work could take is towards weighted graphs and that is exactly what we do in Chapter 5.

The hypothesis test proposed in Article A was shown there to outperform an existing test for the same null hypothesis by Tang et al. (2017). Their test did, however, have a different alternative hypothesis. This is because it builds on fitting an RDPG to each of the graphs letting both U and Λ^k be different under the alternative. A potential criticism of our test and the test by Tang et al.

(2017) is that they require vertex correspondence. Recently there has been more development in this area (Ghoshdastidar and Luxburg, 2018) but theoretical work does exist in tests which do not require this, which unfortunately does not work well in practice (Ghoshdastidar et al., 2017). However, there are many examples of applications where we do have vertex correspondence.

Overall, a new graph model for multiple graphs on the same nodes and an algorithm for fitting said model as well as a hypothesis test for whether two graphs are drawn from the same distribution were proposed in Article A.

Chapter 4

R-package `multiRDPG`

The method for fitting the multiple random dot product graph and hypothesis test proposed in Article A and presented again in Chapter 3 have been implemented in a statistical software package `multiRDPG` for the free open source statistical programming environment R (R Core Team, 2015). The documentation for the package `multiRDPG` is included in Appendix D and we will present the use of this package in this chapter.

4.1 The R-package `multiRDPG`

The `multiRDPG` package consists of two main functions as well as a few utility functions. These two main functions implement Algorithm 1 in the function `multiRDPG` and Algorithm 2 in the function `multiRDPG_test`. In this chapter, these functions are illustrated by working through a simple example. We do this by first creating some simulated data, then fitting the model, and running the test on the data.

We first load the package and generate some simple data. We set the number of nodes $n = 20$, the latent dimension $d = 3$, and the number of graphs $K = 2$. We create a simple orthogonal matrix in `U`.

```
library(multiRDPG)
n <- 20
d <- 3
K <- 2


U <- matrix(0, nrow = n, ncol = d)
U[,1] <- 1/sqrt(n)
U[,2] <- rep(c(1,-1), n/2)/sqrt(n)
U[,3] <- rep(c(1,1,-1,-1), n/4)/sqrt(n)
U
```

```
##             [,1]       [,2]       [,3]
##  [1,] 0.2236068  0.2236068  0.2236068
##  [2,] 0.2236068 -0.2236068  0.2236068
##  [3,] 0.2236068  0.2236068 -0.2236068
##  [4,] 0.2236068 -0.2236068 -0.2236068
##  [5,] 0.2236068  0.2236068  0.2236068
##  [6,] 0.2236068 -0.2236068  0.2236068
##  [7,] 0.2236068  0.2236068 -0.2236068
##  [8,] 0.2236068 -0.2236068 -0.2236068
##  [9,] 0.2236068  0.2236068  0.2236068
## [10,] 0.2236068 -0.2236068  0.2236068
## [11,] 0.2236068  0.2236068 -0.2236068
## [12,] 0.2236068 -0.2236068 -0.2236068
## [13,] 0.2236068  0.2236068  0.2236068
## [14,] 0.2236068 -0.2236068  0.2236068
## [15,] 0.2236068  0.2236068 -0.2236068
## [16,] 0.2236068 -0.2236068 -0.2236068
## [17,] 0.2236068  0.2236068  0.2236068
## [18,] 0.2236068 -0.2236068  0.2236068
## [19,] 0.2236068  0.2236068 -0.2236068
## [20,] 0.2236068 -0.2236068 -0.2236068
```

We also create a list of diagonal matrices in `L`. We let the two diagonal matrices not be equal.

```
L<-list(diag(c(11,6,2)),diag(c(15,4,1)))
L
```

```
## [[1]]
##      [,1] [,2] [,3]
## [1,]   11    0    0
## [2,]    0    6    0
```

```
## [3,]    0    0    2
##
## [[2]]
##      [,1] [,2] [,3]
## [1,]   15    0    0
## [2,]    0    4    0
## [3,]    0    0    1
```

We generate a list of adjacency matrices by sampling from the multi-RDPG model. This is done by creating a matrix `W` representing $W^k = U\Lambda^k U^T$ in iteration k and sampling each element A_{ij}^k of the adjacency matrices independently from a Bernoulli distribution with parameter W_{ij}^k. We make sure that the adjacency matrices are symmetric.

```
A <- list()
for(i in 1:K){
  W <- U%*%L[[i]]%*%t(U)
  A[[i]] <-apply(W,c(1,2),function(x){rbinom(1,1,x)})
  A[[i]][lower.tri(A[[i]])]<-t(A[[i]])[lower.tri(A[[i]])]
}
```

4.1.1 The Function `multiRDPG`

The multi-RDPG model is fitted using the `multiRDPG` function.The function takes in the argument `A` which is a list of adjacency matrices, the dimension of the latent space `d`, the maximum number of iterations that the algorithm can run, `maxiter` (default is 1000), and `tol` which is the tolerance on the convergence criteria (default is 10^{-6}). The convergence criteria is,

$$\texttt{objfun}_{i-1} - \texttt{objfun}_i < \text{tol} \tag{4.1}$$

Where $\texttt{objfun}_i$ is the value of the objective function in (3.3) in the ith iteration.

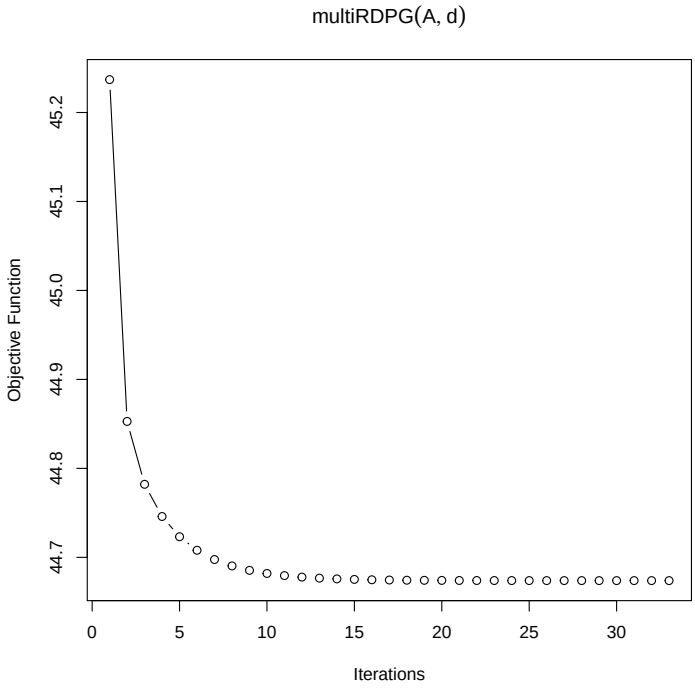
```
fit <- multiRDPG(A,d)
fit


## Call:
## multiRDPG(A = A, d = d)
##
## Number of graphs:  2
```

```
## Converged: Yes
## Number of iterations:  33
##
## Estimated U:
##            [,1]       [,2]       [,3]
## [1,] -0.2235063 -0.2584816  0.1261495
## [2,] -0.2635186  0.1542227  0.2572559
## [3,] -0.2074370 -0.2357472 -0.3685317
## [4,] -0.2387651  0.2029237 -0.2515985
## [5,] -0.2561166 -0.1831481  0.1818533
## [6,] -0.2364721  0.1944138  0.1583812
## [ 14  rows omitted ]
##
## Estimated Lambda:
## [[1]]
##          [,1]     [,2]     [,3]
## [1,] 11.19077 0.000000 0.000000
## [2,]  0.00000 6.900558 0.000000
## [3,]  0.00000 0.000000 3.077524
##
## [[2]]
##          [,1]     [,2]     [,3]
## [1,] 14.94758 0.000000 0.000000
## [2,]  0.00000 4.783966 0.000000
## [3,]  0.00000 0.000000 1.610903
```

The output of the function is an object of the class `multiRDPGfit` which contains the following elements: `U` which is the estimated matrix of joint vectors, `Lambda` which is a list of estimated diagonal matrices, a flag `converged` which is 0 if the algorithm stopped because the maximal number of iterations was reached and 1 if convergence was reached, the number of iterations `iter`, `maxiter` is the given input value, `objfun` is a vector of the values of the objective function in each iteration, the call of the function in `call`, and finally `tol` is the given tolerance. The print method for the class `multiRDPGfit` was created so that it displays the relevant information (as seen above) and then the user can extract the other elements of the list if needed.

Similarly, a plotting method for an object of class `multiRDPGfit` was created. It shows a plot of the values of the objective function in each iteration.

```
plot(fit)
```



4.1.2 The Function multiRDPG_test

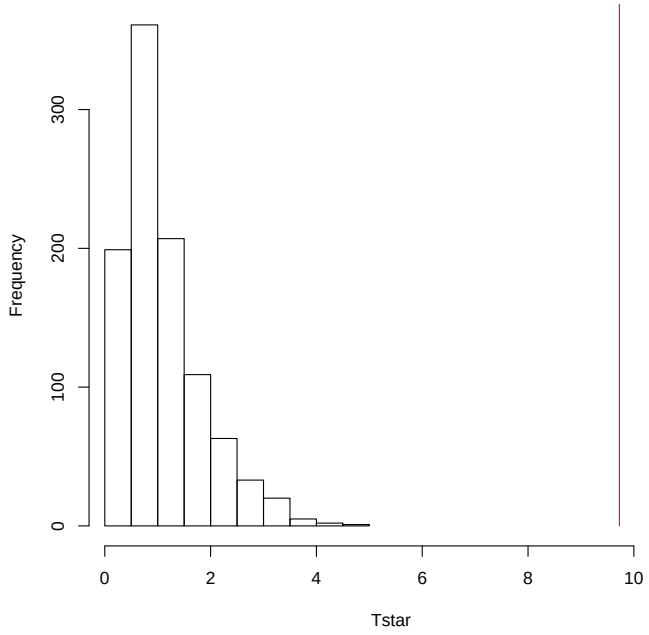
The other main function is the multiRDPG_test which performs the hypothesis test in Algorithm 2. This function again takes all the arguments that multiRDPG takes which are passed on to that function. It also takes the argument B which is the number of permutation iterations. The default is 1000.

```
test_val <- multiRDPG_test(A,d)
test_val



##
##   MultiRDPG Graph Hypotesis Test
##
## data:   A
## t =  9.726736 , p-value =  0 , d =  3
## alternative hypothesis: Lambdas are not equal
```

The output of the function is an object of the class `multiRDPGtest` which contains a list of the following elements: The obtained p-value in `pvalue`, the value of the test statistic in `Tval`, a vector containing all the test statistics for the `B` sets of permuted data in `Tstar`, the model fitted under the null hypothesis in `nullmodel`, the multi-RDPG model fitted under the alternative in `altmodel` and the name of the data in `data.name`. A printing method for an object of class `multiRDPGtest` was created. It prints the name of the data set, the value of the test statistic, the p-value and the dimension d as well as a description of the test performed (as seen above). The rest of the elements can be accessed by the user in the list.

A plotting method for an object of class `multiRDPGtest` was created. It displays a histogram of the null distribution of the test statistic where the value of the test statistic is marked with a vertical red line.

```
plot(test_val)
```



Chapter 5

Generalized Multiple Random Dot Product Graph

In Article B, the multiple random dot product graph model developed in Article A and presented in Chapter 3 is generalized to weighted and undirected graphs. This has been done because weighted graphs often arise, e.g. the Wikipedia data we considered a binary version of in Chapter 3.

In this chapter, we first present the definition of the generalized multiple random dot product graph. We then consider two specific distributions for the edge weights: Poisson and normal. In both cases we present algorithms for performing a hypothesis test for whether two graphs are drawn from the same distribution. Finally, at the end of the chapter we discuss the contributions.

5.1 Definition

In the multiple random dot product graph for unweighted graphs in (3.2) we modeled the probability of an edge between two nodes. This is the same as each

adjacency matrix element following a Bernoulli distribution with the parameter equal to this probability. In order to generalize this model to weighted graphs we changed the Bernoulli distribution to other distributions where we modeled each of the parameters using a product in the same way we previously modeled the probability parameter. The generalized model then is (re-stated from Article B),

$$A_{ij}^k \sim \nu(f_1((\Theta_1^k)_{ij}), \ldots, f_L((\Theta_L^k)_{ij})), \quad \Theta_l^k = U_l \Lambda_l^k U_l^T, \quad k = 1, \ldots, K, \quad (5.1)$$

where $A_{ji} = A_{ij}\ \forall (i,j) \in \{1, \ldots, n\}^2$, ν is a probability distribution with L parameters, U_l for $l = 1, \ldots, L$ are $n \times d$ orthogonal matrices, Λ_l^k for $l = 1, \ldots, L$ and $k = 1, \ldots, K$ are $d \times d$ diagonal matrices with non-negative diagonal elements, $f_l : \mathbb{R} \to S_l$ is a link function and S_l are the domains of the parameters $(\Theta_l^k)_{ij}$ for $l = 1, \ldots, L$ and $k = 1, \ldots, K$. This model is very general but we will investigate two specific cases of it, namely choosing ν to be the Poisson and normal distributions.

5.2 Poisson Distribution

In order to look at the estimation of the model and make a hypothesis test within this framework we considered specific weight distributions. We first considered Poisson distributed weights due to the simplicity of the distribution and because the Poisson distribution often occurs, e.g. in the context of *multigraphs* where multiple edges are allowed between edges. The only parameter of the Poisson distribution is also the mean which is also the case for the Bernoulli distribution which is used in the version of the model (5.1) used in Chapter 3. The generalized multiple random dot product graph model with Poisson distributed edge weights then is (re-stated from Article B),

$$A_{ij}^k \sim \text{Poisson}(f(\Theta_{ij}^k)), \quad \Theta^k = U \Lambda^k U^T, \quad k = 1, \ldots, K \quad (5.2)$$

where $A_{ji} = A_{ij}\ \forall (i,j) \in \{1, \ldots, n\}^2$, U is a $n \times d$ orthogonal matrix , $\Lambda^1, \ldots, \Lambda^K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements and $f : \mathbb{R} \to \mathbb{R}_+$ is a link function.

We will consider the case where we choose the link function $f(x) = \max(0, x)$. This link function is closely related to the identity so for simplicity we will use the identity when fitting the model. The objective function is then the same as in the multiple random dot product graph. This means that we just needed to solve the same optimization problem which is stated in equation (3.3). In Article A, we developed Algorithm 1 for solving this problem and it is also usable here and we therefore used it to fit the model.

5.2.1 Hypothesis Test

Having a method to fit the model, we looked into hypothesis testing in this framework. The hypothesis we wanted to test is,

$$H_0 : \Lambda^1 = \cdots = \Lambda^K. \tag{5.3}$$

which is the same as in Article A.The test statistic also became the same as for the multiple random dot product graph and is stated in (3.6).

The test algorithm has changed as we needed to sample data from Poisson distributions rather than Bernoulli distributions which we sampled from in Algorithm 2. We used parametric bootstrap to obtain the null distribution in Article B instead of the permutation strategy used in Article A. In order to sample using parametric bootstrap, we fit the null model which is the model where the Λs are assumed equal ($\Lambda^k = \Lambda \, \forall k$),

$$A^k_{ij} \sim \text{Poisson}(f(\Theta_{ij})), \quad \Theta = U\Lambda U^T, \quad k = 1, \ldots, K, \tag{5.4}$$

where $A_{ji} = A_{ij} \; \forall (i,j) \in \{1, \ldots, n\}^2$, U is a $n \times d$ orthogonal matrix , Λ is a $d \times d$ diagonal matrix with nonnegative diagonal elements and $f : \mathbb{R} \to \mathbb{R}_+$ is a link function. We compute the test statistic (3.6) by fitting the null model in (5.4) and the alternative in (5.2) as described in Article B. We then repeatedly generate data from the null model and calculate the test statistic on the simulated data to obtain a null distribution for the test statistic such that we can compute a p-value. The test algorithm is re-stated in Algorithm 3.

In Article B, we performed simulation studies which showed that under the null hypothesis the empirical distribution of the p-values are satisfactorily close to uniformly distributed indicating adequate type I error control. Further, we showed that the fraction of rejected hypotheses increases as the diagonal matrices become more different from each other as well as with the size of the graphs indicating that the test has adequate power.

5.2.2 Example: Wikipedia Data

We again consider the Wikipedia example from Sections 2.1.1, 3.1.2, and 3.2.1.

We have previously constructed the graphs by letting there be an edge between two nodes if either article links to the other. But the articles might actually link to the same other article more than once (or both might link to each other) making it a multigraph which can be represented as a weighted graph. Letting

Algorithm 3 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (5.2) (re-stated from Article B).

1. Fit the model in (5.4) by finding the eigen decomposition $\frac{1}{K}\sum_{k=1}^{K} A_+^k = QDQ^T$ where the diagonal elements of D are in non-increasing order. Then let $\hat{U}$ be the first d columns of Q and $\hat{\Lambda}$ contain the first d diagonal values of Q
2. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (3.6).
3. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{k,*b}\right)_{ij}$ be generated by simulating from the distribution Poisson$(f((\hat{U}\hat{\Lambda}\hat{U}^T)_{ij}))$ for all $k = 1, \ldots, K$
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (3.6).
4. Compute the p-value,

$$p = \frac{1}{B}\sum_{b=1}^{B} I_{T\left(A_+^{1,*b},\ldots,A_+^{K,*b}\right) \geq T\left(A_+^1,\ldots,A_+^K\right)},$$

where I_C is an indicator function that equals one if the event C holds, and equals zero otherwise.

the number of links be the edge weights, a subset of the English adjacency matrix is,

$$A^{\text{English}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 2 & 0 & 0 & 0 & 3 & 1 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 2 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix},$$

and we similarly display a subset of the French adjacency matrix,

$$A^{\text{French}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & 3 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

If we compare these matrices to the binary adjacency matrices in Section 2.1.1, we see that there are edges in the same places but that some of them actually have a weight higher than one.

Because this is count data (number of links), the natural distribution for the weights is the Poisson distribution. This is the reason that this example was also included in Article B. In this case we did not downsample the English graph but rather scaled all elements of the adjacency matrix such that the mean of all elements in the English matrix is equal to the mean of all elements in the French matrix. We fit the model (5.2) to this data with $d = 5$ and get,

$$\hat{\Lambda}^{\text{English}} = \begin{bmatrix} 147.0 & 0 & 0 & 0 & 0 \\ 0 & 133.9 & 0 & 0 & 0 \\ 0 & 0 & 61.4 & 0 & 0 \\ 0 & 0 & 0 & 110.9 & 0 \\ 0 & 0 & 0 & 0 & 29.6 \end{bmatrix},$$

$$\hat{\Lambda}^{\text{French}} = \begin{bmatrix} 204.5 & 0 & 0 & 0 & 0 \\ 0 & 10.5 & 0 & 0 & 0 \\ 0 & 0 & 78.3 & 0 & 0 \\ 0 & 0 & 0 & 26.4 & 0 \\ 0 & 0 & 0 & 0 & 71.5 \end{bmatrix}.$$

Now testing whether these two matrices can be said to be equal we run the hypothesis test in Algorithm 3. This gives a test statistic of 11018.8 and a p-value of $p = 0$. And we reject the hypothesis that the two are equal.

5.3 Normal Distribution

We consider another example of the edge weight distribution in Article B namely the normal distribution. In this section, we do not strictly follow the definition in (5.1) because we only estimate the mean parameter using a matrix product and we let the variance be equal for all elements across graphs and let the correlation between the matrix elements be zero. The model in this case is (restated from Article B),

$$A_{ij}^k \sim \text{Normal}(f(M_{ij}^k), \sigma^2), \quad M^k = U\Lambda^k U^T, \quad \sigma^2 \geq 0, \quad k = 1, \ldots, K \tag{5.5}$$

where $A_{ji} = A_{ij}$ $\forall (i,j) \in \{1, \ldots, n\}^2$ and $f : \mathbb{R} \to \mathbb{R}$ is a link function.

We let the link function f be the identity and then to estimate the mean parameter we again get the optimization problem in (3.3) which again is solvable using Algorithm 1. For now we do not worry about estimating the variance parameter but we will need to estimate it under the null hypothesis.

5.3.1 Hypothesis Test

The test hypothesis of whether two (or more) graphs are drawn from the same distribution again is,

$$H_0 : \Lambda^1 = \cdots = \Lambda^K \tag{5.6}$$

as the variance is assumed equal across graphs. The test statistic is also the same as in equation (3.6). In order to test the hypothesis (5.6), we needed to be able to sample from the model under the null hypothesis. That is from,

$$A_{ij}^k \sim \text{Normal}(f(M_{ij}), \sigma^2), \quad M = U\Lambda^k U^T, \quad \sigma^2 \geq 0, \quad k = 1, \ldots, K \tag{5.7}$$

where $A_{ji} = A_{ij}$ $\forall (i,j) \in \{1, \ldots, n\}^2$ and $f : \mathbb{R} \to \mathbb{R}$ is a link function (here the identity). The mean parameter can again be estimated using the eigenvalue decomposition. However, we also needed to estimate the variance parameter. It can be estimated as the variance in a multidimensional normal distribution with equal variance and no correlations, and so the unbiased variance estimate

is (from Article B),

$$\tilde{\sigma}^2 = \left(\frac{K}{2}(n^2+n) - nd + \frac{d(d-1)}{2}\right)^{-1} \sum_{k=1}^{K}\sum_{i=1}^{n}\sum_{j=1}^{i}(A_{ij}^k - \hat{M}_{ij})^2 \tag{5.8}$$

because $\frac{K}{2}(n^2+n)$ is the number of observations and $nd + \frac{d(d-1)}{2}$ is the number of parameters estimated in the mean $\hat{M}$. So it is the standard estimator which is corrected with the number of parameters estimated in the mean in order to obtain an unbiased estimate.

We created a null distribution for the test statistic using parametric bootstrap sample from the model under the null hypothesis (5.7). The algorithm is restated in Algorithm 4.

In Article B we demonstrated through simulation studies that the p-values are uniformly distributed under the null hypothesis indicating adequate type I error control. We also demonstrated that the fraction of rejected hypotheses increases as the graphs become more different from each other as well as with the size of the graphs.

5.3.2 Example: Oribatid Mites

We consider an example of data which has approximately normally distributed edge weights. The data consist of counts of different species of Oribatid mites in 70 soil cores from different locations (Borcard et al., 1992; Borcard and Legendre, 1994, 2002; Oksanen et al., 2017). There are 35 mite species counted in the cores. We construct a network by letting the locations be the nodes and the edge weights be given as the difference in the population composition in the two corresponding locations in terms of the Bray-Curtis dissimilarity (Bray and Curtis, 1957),

$$BC_{ij} = \frac{\sum_s |y_{si} - y_{sj}|}{\sum_s (y_{si} + y_{sj})}, \tag{5.9}$$

where y_{si} is the number of species s present at sample site i.

We split the mite species into two groups (the first 17 species and then the remaining 18) and construct a network for each group. We fit the model to these two networks with $d = 2$ and get that the estimated diagonal values are,

$$\hat{\Lambda}^1 = \begin{bmatrix} 43.22 & 0 \\ 0 & 0.13 \end{bmatrix}, \quad \hat{\Lambda}^2 = \begin{bmatrix} 44.99 & 0 \\ 0 & 0.12 \end{bmatrix}.$$

Algorithm 4 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (5.5) (re-stated from Article B).

1. Estimate the mean parameter in (5.7) by finding the eigendecomposition $\frac{1}{K}\sum_{k=1}^{K} A_+^k = QDQ^T$ where the diagonal elements of D are in non-increasing order. Then let $\hat{U}$ be the first d columns of Q and $\hat{\Lambda}$ contain the first d diagonal values of Q
2. Estimate the variance parameter in (5.7) according to (5.8)
3. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (3.6).
4. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{k,*b}\right)_{ij}$ be generated by simulating from the distribution Normal$((\hat{U}\hat{\Lambda}\hat{U}^T)_{ij}, \tilde{\sigma}^2)$ for all $k = 1, \ldots, K$
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (3.6).
5. Compute the p-value,

$$p = \frac{1}{B}\sum_{b=1}^{B} I_{T\left(A_+^{1,*b},\ldots,A_+^{K,*b}\right) \geq T\left(A_+^1,\ldots,A_+^K\right)},$$

where I_C is an indicator function that equals one if the event C holds, and equals zero otherwise.

These two matrices seem to have a similar structure but we might want to know whether they can be said to be equal. In Article B, we test the hypothesis $H_0 : \Lambda^1 = \Lambda^2$ using Algorithm 4. We get the value of the test statistic to $T(A^1_+, A^2_+) = 1.55$ and a p-value of $p = 0.037$ meaning that we reject the hypothesis that they are equal. We will not make any biological interpretations of this.

As a point of comparison we create two groups randomly. We sample 17 species to go in one group so that the first group contains the species with the indices $\{1, 6, 7, 14, 19, 20, 21, 22, 23, 26, 27, 28, 29, 30, 31, 32, 33\}$ and the second group contains the remaining species. We again construct two networks and fit the model. This time we get the following estimated diagonal values,

$$\hat{\Lambda}^{\text{group 1}} = \begin{bmatrix} 44.23 & 0 \\ 0 & 0.12 \end{bmatrix}, \quad \hat{\Lambda}^{\text{group 2}} = \begin{bmatrix} 43.04 & 0 \\ 0 & 0.13 \end{bmatrix}$$

which by just looking at the values are slightly more similar. We test the hypothesis that they are equal using Algorithm 4 and get a value of the test statistic to be $T(A_+^{\text{group 1}}, A_+^{\text{group 2}}) = 0.71$ and a p-value of of $p = 0.34$ so we cannot reject the hypothesis that they are equal. We again refrain from making biological interpretations.

5.4 Discussion

The generalized multi-RDPG is presented in this chapter. It models multiple weighted random graphs on the same set of nodes. We especially consider two versions of the model with Poisson and normally distributed edge weights. In the latter, we only consider the simple version of the distribution where all elements across graphs have equal variance and are uncorrelated. This is a simple version and one could consider modeling the variance with a matrix product as well.

In both the Poisson and normal cases, we consider one link function such that we can use the identity in the optimization problem and thereby use the algorithm proposed in article A for fitting. One could consider other link functions which would change the optimization problem.

We developed algorithms for testing whether two graphs are drawn from the same distribution. These were developed in the cases of Poisson and normally distributed edge weights. A potential criticism of these tests is that they require vertex correspondence.

Overall, we presented a generalization of the multi-RDPG and tests for whether

two graphs are drawn from the same distribution in the cases of Poisson and normally distributed edge weights.

Part II

Chapter 6

Disease Prediction

The second part of this thesis concerns prediction of disease progression. In Article C, we reviewed approaches for predicting diabetes progression and then compared methods from each of these approaches by applying them to a diabetes data set. In this chapter, we consider what disease prediction is and why it is of interest as well as review the three common approaches we found in literature.

6.1 Disease Prediction

In prediction of disease progression the aim is to be able to predict how a specific patient's disease evolves, e.g. how long it will take for them to move on to the next disease stage. This is of interest because patient trajectories can be very particular to the individual and it is desirable for the patient and the clinic to have precise predictions of changes. This has also led to a focus on disease prediction because it can have an influence on clinical decisions. So while doctors have always been tailoring treatments to the individual, one aim of disease prediction is to make it possible to do this more precisely and automatically based on increasing amounts of available data and computing resources.

6.2 Three Common Approaches

In Article C, we outlined three main approaches to disease prediction which we found in diabetes literature. The first two use survival models, i.e. they directly model the time until an event happens. We split this into two because the first is a standard statistical approach using well known parametric or semi-parametric methods. The second approach is to use machine learning methods developed for survival data. These are newer approaches to the same problem which aim to solve some problems present in the standard methods. The third approach is to change the time-to-event outcome into a classification outcome. We then predict whether the event happens before or after a pre-specified cut-off, t_c. The outcome then becomes $Y_i = I(T_i < t_c \wedge \delta_i = 1)$ where $I(\cdot)$ is an indicator function. This can be done to make translation into the clinic easier because the outcome is easy to understand.

6.2.1 Survival Analysis

Both the first and the second approach use survival models. In survival analysis the response variable is the time until some event occurs (Harrell Jr, 2018). The response is continuous but is allowed to be incompletely observed. That means for some subjects we do not observe the event because it does not happen within the study period or the subject drops out of the study before it happens. In this case, this subject is said to be *right censored*, i.e. we only know that the time-to-event is greater than the censoring time (see Figure 6.1). We let T_i be the time until event (or the subject left the study) for subject i and δ_i an indicator of whether the event happened or not.

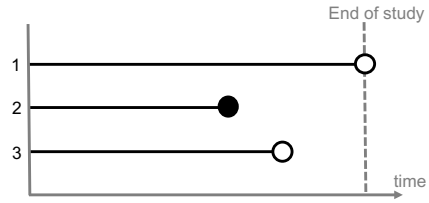


Figure 6.1: Example of survival data where ∘ indicates censoring and • indicates an event. Here subject 1 is censored at the end of the study, subject 2 has the event happening, and subject 3 is censored before the end.

Subjects can also be left censored, i.e. entering the study late, but this kind of censoring is not present in the comparison study in Article C.

The first approach uses a parametric or semi-parametric model to directly model the time-to-event. The methods in this category are for example accelerated failure time models and proportional hazards models of which *Cox proportional hazards model* is the most commonly used and the one we chose to represent this approach in Article C (Cox, 1972). Proportional hazard means that we model the hazard (rate of events), λ, at time t as proportional to some base hazard. In Cox models the base hazard λ_0 is unknown and the model is,

$$\lambda(t|X_i) = \lambda_0(t)\exp(X_i\beta) \tag{6.1}$$

where X_i are the explanatory variables of subject i.

The second approach uses machine learning methods to model the time-to-event. This includes decision trees and random forest models. In Article C, we chose the conditional inference forest by Hothorn et al. (2004) to represent this approach because it has been shown to outperform competing random forest methods (Nasejje et al., 2017). This model takes a random forest (Breiman, 2001) approach and builds trees on bootstrap samples of the data. Each tree is built by recursively repeating two steps: First, testing the global hypothesis of independence between any of the variables and the response, and choosing the variable with the strongest association. Second, splitting the sample space of the chosen variable into two disjoint sets which define the child nodes.

6.2.2 Classification

The third approach changes the time-to-event problem to a classification problem. We split the outcome at a pre-specified cut-off and classify whether the event happens before the cut-off or not (Figure 6.2).

Both linear models and machine learning methods can be used for prediction in classification problems. In Article C, we chose to use a neural network to represent this approach because it is widely used (Venables and Ripley, 2002). We chose a simple neural network with a single hidden layer in addition to the input and output layers . Each layer consists of a number of neurons which add together their inputs and apply an activation function. The model here is,

$$\hat{y} = \phi_0\left(\alpha + \sum_h w_h\phi_h\left(\alpha_h + \sum_i w_{ih}X_i\right)\right) \tag{6.2}$$

where ϕ_0 is the output layer activation function, ϕ_h is the hidden layer activation function, w_h and w_{ih} are weights, α and α_h are constants, h runs over the hidden units, and i runs over the observations.

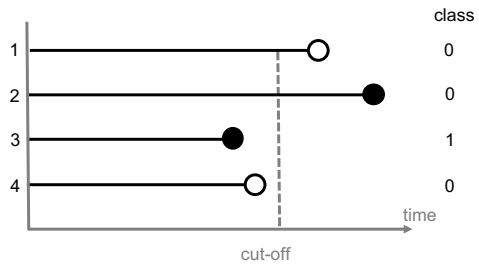


Figure 6.2: Example of changing survival data to classification data. ○ indicates censoring and • indicates an event. Here subject 1 and 4 are censored so they have class 0, subject 2 has the event happening after the cut-off so it has class 0, and subject 3 has the event happening before the cut-off so it has class 1.

In order to be able to compare the third approach to the others we wanted to dichotomize the predicted survival time after predicting. This is, however, not possible with the Cox model because it is a proportional hazards model and only gives predictions relative to an unknown base hazard. This means we can compare the first and second approach in a survival setting, and the second and third in a classification setting.

Chapter 7

Comparison Study on Diabetes Data

In Chapter 6, we outlined three common approaches to disease prediction that we found in Article C and chose a method within each approach.

In this chapter, we consider a diabetes data set in which type 2 diabetes patients have been followed for more than 20 years that we analysed in Article C (Doney et al., 2004, 2005; Zhou et al., 2014). From a biological perspective, the goal of analyzing this data is to be able to predict when the patients will start the final type of diabetes treatment - insulin treatment. Predicting the time until a patient goes on to insulin treatment is a time-to-event or survival modeling problem. We compare the three approaches outlined in Chapter 6 by applying the methods chosen within each approach to this data set. This lets us compare the three methods in a way that is true to the challenges faced by researchers working in the field which is the goal seen from our perspective.

In Article C, we focused on the statistical and machine learning methods and evaluated the pros and cons of the methods based on the data analysis. Here we first present the data set, then we consider the way we evaluated the models, give the conclusions, and finally, discuss the contributions.

7.1 Data

We start by presenting the data set used in Article C which contains the medical records of patients with type 2 diabetes from the Genetics Diabetes Audit and Research (GoDARTS) database (Doney et al., 2004, 2005; Zhou et al., 2014). We cleaned the data in various ways ending up with 6324 patients. The data set contains many sources of data. It contains the following clinical variable types: Anthropometric data such as height and weight, life-style data, social class derived from the Scottish Index of Multiple Deprevation, drug prescriptions, and diagnosis details. These are of interest because type 2 diabetes is a life-style disease. A detailed list of variables included in the models is given in Article C.

The data also includes longitudinal measurements of biochemical variables such as cholesterol and glycated haemoglobin (HbA1c). These variables are measured irregularly throughout the study period. In Figure 7.1 the HbA1c measurements throughout the study period for all individuals are shown. The colors indicate the values at diagnosis. We notice that before diagnosis all values are below 6.5% which is natural as the diagnosis is confirmed when HbA1c$>$ 6.5%. HbA1c is an important marker of diabetes, because it helps identify blood sugar levels. However, there is still no obvious pattern seen in Figure 7.1 which illustrates the complexity of the data.

We have utilized just the data from one year around diagnosis as we are looking at the prediction of the time to insulin treatment at the time of diagnosis. See Article C for details about the variables.

The response variable in the data is the time from confirmed diagnosis until insulin treatment (or until the patient left the study) as well as an indicator of whether the patient received insulin. This outcome is also split into classification problems as described in Section 6.2. We used cut-offs of 1, 3, and 5 years.

7.1.1 Feature Extraction

From the biochemical longitudinal variables we extract features in two ways so that we have two versions of the data set. The first version is simply the measurements closest to the diagnosis and within six months of the diagnosis (six months before to six months after diagnosis). This gives one value per biochemical variable for each subject. The second version of the data set is constructed by modeling the linear trend and first order auto-regression of all the measurements in the year around diagnosis for each biochemical variable

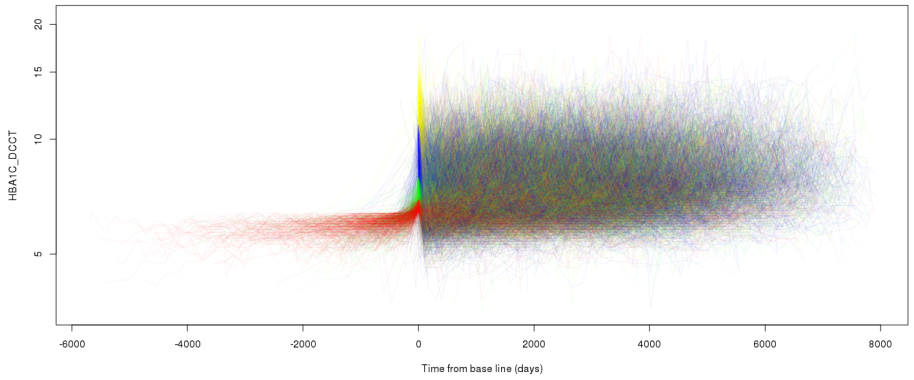


Figure 7.1: The HbA1c (in percent) levels for all patients over the full time period where day zero is the day of diagnosis. The colors indicate their value at diagnosis: Colors indicate value at diagnosis. Black: <6.5, red: 6.5-7, green: 7-8, blue: 8-11, and yellow: >11. This figure has previously been used in Nielsen et al. (2018).

(Eckner, 2012; Mudelsee, 2014).This feature extraction is complicated by having irregular time series as well as in some cases having few observations in the year. The parameters of the time series models were extracted to use as variables for the survival prediction models which gives three features extracted for each biochemical variable in the second version of the data set.

There are two versions of the data set that contain the same outcome and clinical variables but the features extracted from the biochemical measurements are the same.

7.2 Model Evaluation

We need performance measures to evaluate the prediction performance of the three methods outlined in Section 6.2 when applied to the two versions of the data set constructed as in Section 7.1.1. The models and their appropriateness for different tasks are evaluated using their prediction performance, where we remember that we work with two versions of the outcome (Cox model and conditional inference forest), but we also considered how well they handled common data challenges.

7.2.1 Performance Measures

The prediction performance of the models with survival outcome was evaluated using concordance indices by Harrell Jr et al. (1982) and Uno et al. (2011). These evaluated the discriminative powers of the survival models where the latter takes the amount of censoring into account. The survival prediction performance was also evaluated using the integrated Brier score (Graf et al., 1999). The prediction performances of the classification models were evaluated using the accuracy, sensitivity, specificity, and Matthews' correlation coefficient Matthews (1975). The last was chosen because it can handle imbalanced data. For all measures we have used cross-validation to obtain an estimate of the generalization error of the methods. The results of the analysis are summarized in Tables 3 and 4 in Article C. But in general the Cox model and the conditional inference forest perform similarly but the Cox model performs slightly better in the survival setting. In the classification setting the neural network outperformed the conditional inference forest.

7.2.2 Data Challenges

Besides the prediction performance we were also interested in how well the methods handled common data challenges which is why we conducted the analysis on a real data set. In the study in Article C, we encountered some of these common data challenges, the first being missing data. The conditional inference forest could natively handle missing data through surrogate splits meaning it partitions on a different variable that leads to the same split (Hothorn et al., 2010). The two other methods, Cox model and neural network (in the implementation we used (Ripley and Venables, 2016)), could not handle missing data and we had to either discard data or impute. This introduced another choice of what to do in two of the approaches.

Another challenge we met was data imbalance, i.e. the number of observations in each class not being equal. In the survival data this was present if interpreting whether the subject is censored as a class as we have 58% censoring. For the dichotomized response, there were many more observations that did not go on to insulin before the specified cut-offs (1, 3 or 5 years) than there did and the data was highly imbalanced. This was especially true using the year 1 cut-off and then less for later years, but the data was still imbalanced using the year 5 cut-off. In both survival and the classification settings, we dealt with the imbalance using downsampling. We found that it was important to deal with the imbalance for the performance of the classification models but that balancing censoring did not have a large impact on the performance of the survival predictions.

7.3 Conclusions

As conclusions to the comparison study in Article C we found that there are a few areas to especially consider when choosing the best method for the analysis. With regards to data imbalance, we found that it was very important to deal with the imbalance for classification problems. This is well known and many methods exist for it, e.g. downsampling. Downsampling the imbalance of the censoring in the survival analysis did, however, not have a large effect.

We found that it is important to consider whether the interpretation of the model is of interest when choosing the method. If it is important, then a simple method might be preferable. Finally, we found that methods trained directly to a task performed the best. This means that if there is an interest in a dichotomized response then it is preferable to directly build a classification model rather than dichotomize the survival predictions post-training.

7.4 Discussion

In Article C, we compared three approaches for predicting disease progression. On the basis of our study we gave a set of recommendations. These came both from the literature study and the analysis of the GoDARTS data set performed in the article.

The results in Article C come from analysis of a real data set. This means that we do not know the true relation between the outcome and the explanatory variables. This gives a realistic comparison of methods but also a less controlled comparison than if we had used simulated data. The low level of control of the properties of the data set could be a disadvantage but it also lets the comparison be true to real challenges.

Furthermore we had chosen not to use the same strategies for handling missing data for all the methods, meaning that the difference in the results is not due to one difference in the approach. This was done because the different methods call for different approaches for handling missing data. The conditional inference forest can for example handle it as part of the method.

Finally, we consider both the time-to-event prediction approaches and the dichotomizing approach. From the literature study we found that dichotomizing is generally not advisable due to the loss of information (Fedorov et al., 2009). However, the outcome is of interest to practitioners because of the simplicity.

We found through the data analysis that it was then best to dichotomize before training the method rather than after.

Chapter 8

Conclusion

In this thesis, we considered statistical methods for learning from biological and medical data. We considered methods applicable to specific problems that we have observed or encountered.

In the first part of the thesis, we developed methods for modeling multiple graphs. We proposed the multiple random dot product graph in Article A. This model came from a direct generalization of the random dot product graph (Nickel, 2008; Scheinerman and Tucker, 2010). The model gave a way of representing a distribution of multiple graphs on the same set of nodes by letting the graphs have common latent vectors but a different set of latent values for each graph. In this model, one can interpret the graphs as being embedded in a space and having different coordinates in that space. In Article A, we gave an algorithm for fitting the model where we could fit all latent dimensions simultaneously when the link function was chosen to be the identity. This gave improved empirical results. Within the framework of this model we also presented an approach for testing whether two graphs are drawn from the same distribution. Through simulation studies in Article A, we showed indication that the test controlled type I error and had adequate power under the alternative.

The methods for fitting the multiple random dot product graph model and performing the test were implemented in the statistical programming language R (R Core Team, 2015) in the package `multiRDPG` (see Appendix D) which is

available on the Comprehensive R Archive Network (CRAN).

We generalized the multiple random dot product graph to weighted networks in Article B so that we can represent the distributions of multiple weighted graphs on the same set of nodes. We specifically considered the Poisson distribution and the normal distribution with equal variance as distributions of the edge weights. In these two cases, we found that the models could be fitted using the algorithm developed in Article A. We also presented algorithms for testing whether two (or more) graphs were drawn from the same distribution in these two cases where in the normal distribution we had to find an unbiased estimate of the variance.

In short, in the first part of this thesis we have presented a model for representing the distributions of multiple graphs on the same set of nodes, an algorithm for fitting the models, and hypothesis tests within that framework for whether two (or more) graphs were drawn from the same distribution. This has been developed for both unweighted and weighted graphs.

In the second part of this thesis, we considered methods for predicting disease progression. In Article C, we found three common approaches in diabetes literature. The first was traditional statistical methods for survival analysis such as the Cox proportional hazards model; the second was newer developments for survival data such as the conditional inference forest; and the third was machine learning methods for the dichotomized response such as a neural network. We compared the methods by applying them to a diabetes data set. We found that the Cox model and the conditional inference forest performed similarly in terms of prediction performance but each have their own advantage in other aspects. The conditional inference forest can directly give a prediction of the survival time which is not possible in the Cox model as it has an unknown base hazard. The Cox model is, however, easier to interpret. The predictions of the conditional inference forest can be compared to the dichotomized predictions from the neural network. In this comparison we found that the neural network performed best.

Overall, in part two we found that the methods are not direct substitutes for each other and so the goal of the analysis is important when choosing a method.

8.1 Future Work

In the multiple random dot product graph and in the generalized version, we have only considered the link function to be the identity when fitting the models.

Other link functions might be more natural in some cases. The logistic function $f(x) = \exp(x)/(1 + \exp(x))$ might be a more natural choice in the model for unweighted graphs because it actually maps from $\mathbb{R}$ to $[0, 1]$. This would require a modification of the fitting algorithm.

It would also be interesting to investigate how to optimally choose the latent dimension d. This is a problem that often shows up when finding low dimensional representations of data. Another interesting aspect to investigate is interpretations of the latent dimensions. Because of the restrictions of the U and Λ^k the graphs can be interpreted as lying in a d-dimensional space. It would be interesting to investigate the meanings of these dimensions.

Estimation of the variance in the model with normally distributed weights could also be interesting to investigate further. The current estimation is unbiased in the limit but the estimate is not useful for small graphs. It could therefore be interesting to further develop this work for small graphs.

Another direction of future work could be to investigate a proportional hypothesis as a way of formalizing the downsampling we have done in the examples. This idea has come up in discussions while writing Article A and the hypothesis would be $H_0 : \Lambda^1 \propto \Lambda^2$. In the unweighted model this would mean that we are also interested in the case where the probabilities of an edge are proportional to each other but otherwise have the same structure. This would require a modification of the testing algorithm as the permutation method is not suitable for this hypothesis.

One could also consider different probability distributions as the edge weight distributions. For example, under the normal distribution we will always have all the edges. This is appropriate in the Oribatid mites data. There might, however, be examples where it is inappropriate but where the edges that are still present have normally distributed weights. In this case, it might be more appropriate to have both a distribution for whether there is an edge and then for the edges to have a weight distribution, i.e. the product between the Bernoulli distribution and for example the normal distribution.

Future work in the second part of the thesis could be to apply the same analysis to a different data set to investigate if we reach the same conclusions. It could also be interesting to repeat parts of the analysis in a simulation study such that all aspects of the data set are controlled.

Besides this, we have only chosen one method to represent the approach. Other choices of methods could be interesting to explore. For example, we have chosen to represent the first approach by the Cox proportional hazards model which cannot give direct prediction of the survival time because the base hazard is

unknown. However, other proportional hazards models which estimate the base hazard exist. These could be an interesting alternative choice.

Bibliography

Aicher, C., A. Z. Jacobs, and A. Clauset (2013). Adapting the stochastic block model to edge-weighted networks. *arXiv preprint arXiv:1305.5782*.

Bastian, M., S. Heymann, and M. Jacomy (2009). Gephi: An open source software for exploring and manipulating networks. In *International AAAI Conference on Weblogs and Social Media*, pp. 361–362. AAAI.

Boland, P. J. (1984). A biographical glimpse of William Sealy Gosset. *The American Statistician 38*(3), 179–183.

Borcard, D. and P. Legendre (1994). Enviromental control and spatial structure in ecological comminities: an example using Oribatid mites (Acari,Oribatid). *Enviromental and Ecological Statistics 1*, 37–61.

Borcard, D. and P. Legendre (2002). All-scale spetial analysis of ecological data by means of principal coordinates of neighbour matrices. *Ecological Modelling 153*, 51–68.

Borcard, D., P. Legendre, and P. Drapeau (1992). Partialling out the spatial component of ecological variation. *Ecology 73*, 1045–1055.

Bray, J. R. and J. Curtis (1957). An ordination of upland forest communities of southern Wisconsin. *Ecological Monographs 27*, 325–349.

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological) 34*(2), 187–220.

DeFord, D. R. and D. N. Rockmore (2016). A random dot product model for weighted networks. *arXiv preprint arXiv:1611.02530*.

Doney, A. S., B. Fischer, G. Leese, A. D. Morris, and C. N. Palmer (2004). Cardiovascular risk in type 2 diabetes is associated with variation at the pparg locus: a go-darts study. *Arteriosclerosis, thrombosis, and vascular biology 24*(12), 2403–2407.

Doney, A. S., S. Lee, G. P. Leese, A. D. Morris, and C. N. Palmer (2005). Increased cardiovascular morbidity and mortality in type 2 diabetes is associated with the glutathione s transferase theta–null genotype: A go-darts study. *Circulation 111*(22), 2927–2934.

Dong, X., P. Frossard, P. Vandergheynst, and N. Nefedov (2014). Clustering on multi-layer graphs via subspace analysis on grassmann manifolds. *IEEE Transactions on Signal Processing 62*(4), 905–918.

Durante, D., D. B. Dunson, and J. T. Vogelstein (2017). Nonparametric Bayes modeling of populations of networks. *Journal of the American Statistical Association 112*(520), 1516–1530.

Eckner, A. (2012). A note on trend and seasonality estimation for unevenly-spaced time series.

Erdös, P. and A. Rényi (1959). On random graphs, i. *Publicationes Mathematicae (Debrecen) 6*, 290–297.

Fedorov, V., F. Mannino, and R. Zhang (2009). Consequences of dichotomization. *Pharmaceutical Statistics: The Journal of Applied Statistics in the Pharmaceutical Industry 8*(1), 50–61.

Garlaschelli, D. (2009). The weighted random graph model. *New Journal of Physics 11*(7), 073005.

Ghoshdastidar, D. and U. v. Luxburg (2018). Practical methods for graph two-sample testing. In *Conference on Neural Information Processing Systems*.

Ghoshdastidar, D., U. von Luxburg, M. Gutzeit, and A. Carpentier (2017). Two-sample tests for large random graphs using network statistics. *arXiv preprint arXiv:1705.06168*.

Graf, E., C. Schmoor, W. Sauerbrei, and M. Schumacher (1999). Assessment and comparison of prognostic classification schemes for survival data. *Statistics in medicine 18*(17-18), 2529–2545.

Harrell Jr, F. E. (2018). *rms: Regression Modeling Strategies*. R package version 5.1-2.

Harrell Jr, F. E., R. M. Califf, D. B. Pryor, K. L. Lee, R. A. Rosati, et al. (1982). Evaluating the yield of medical tests. *Jama 247*(18), 2543–2546.

Hermundstad, A. M., D. S. Bassett, K. S. Brown, E. M. Aminoff, D. Clewett, S. Freeman, A. Frithsen, A. Johnson, C. M. Tipper, M. B. Miller, et al. (2013). Structural foundations of resting-state and task-based functional connectivity in the human brain. *Proceedings of the National Academy of Sciences 110*(15), 6169–6174.

Hoff, P. (2008). Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, pp. 657–664.

Hoff, P. D., A. E. Raftery, and M. S. Handcock (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association 97*(460), 1090–1098.

Holland, P. W., K. B. Laskey, and S. Leinhardt (1983). Stochastic blockmodels: First steps. *Social Networks 5*(2), 109–137.

Hothorn, T., K. Hornik, C. Strobl, and A. Zeileis (2010). Party: A laboratory for recursive partytioning.

Hothorn, T., B. Lausen, A. Benner, and M. Radespiel-Tröger (2004). Bagging survival trees. *Statistics in medicine 23*(1), 77–91.

Ma, Z. and Z. Ma (2017). Exploration of large networks via fast and universal latent space model fitting. *arXiv preprint arXiv:1705.02372*.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure 405*(2), 442–451.

Mudelsee, M. (2014). *Climate time series analysis*. Springer.

Nasejje, J. B., H. Mwambi, K. Dheda, and M. Lesosky (2017). A comparison of the conditional inference survival forest model to random survival forests based on a simulation study as well as on two applications with time-to-event data. *BMC medical research methodology 17*(1), 115.

Nickel, C. L. M. (2008). *Random Dot Product Graphs A Model For Social Networks*. Ph. D. thesis, The Johns Hopkins University.

Nielsen, A. M., R. L. Nielsen, L. Donnelly, K. Zhou, E. Pearson, A. B. Dahl, R. Gupta, and B. K. Ersbøll (2018). Prediction of disease progression using irregularly sampled longitudinal data. Poster presented at the Machine Learning Summer School (Buenos Aires, Argentina).

Oksanen, J., F. G. Blanchet, M. Friendly, R. Kindt, P. Legendre, D. McGlinn, P. R. Minchin, R. B. O'Hara, G. L. Simpson, P. Solymos, M. H. H. Stevens, E. Szoecs, and H. Wagner (2017). *vegan: Community Ecology Package*. R package version 2.4-5.

O'Connor, L., M. Médard, and S. Feizi (2017). Maximum likelihood latent space embedding of logistic random dot product graphs. *arXiv preprint arXiv:1510.00850*.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Ripley, B. D. and W. N. Venables (2016). *nnet: Feed-Forward Neural Networks and Multinomial Log-Linear Models*. R package version 7.3-12.

Scheinerman, E. R. and K. Tucker (2010). Modeling graphs using dot product representations. *Computational Statistics 25*(1), 1–16.

Shiga, M. and H. Mamitsuka (2012). A variational Bayesian framework for clustering with multiple graphs. *IEEE Transactions on Knowledge and Data Engineering 24*(4), 577–590.

Student (1908). The probable error of a mean. *Biometrika*, 1–25.

Suwan, S., D. S. Lee, R. Tang, D. L. Sussman, M. Tang, C. E. Priebe, et al. (2016). Empirical Bayes estimation for the stochastic blockmodel. *Electronic Journal of Statistics 10*(1), 761–782.

Tang, M., A. Athreya, D. L. Sussman, V. Lyzinski, Y. Park, and C. E. Priebe (2017). A semiparametric two-sample hypothesis testing problem for random graphs. *Journal of Computational and Graphical Statistics 26*(2), 344–354.

Tang, R. (2017). *Robust Estimation from Multiple Graphs*. Ph. D. thesis, Johns Hopkins University.

Tang, W., Z. Lu, and I. S. Dhillon (2009). Clustering with multiple graphs. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pp. 1016–1021. IEEE.

Uno, H., T. Cai, M. J. Pencina, R. B. D'Agostino, and L. Wei (2011). On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in medicine 30*(10), 1105–1117.

Venables, W. N. and B. D. Ripley (2002). *Modern Applied Statistics with S* (Fourth ed.). New York: Springer. ISBN 0-387-95457-0.

Wang, L., Z. Zhang, and D. Dunson (2017). Common and individual structure of multiple networks. *arXiv preprint arXiv:1707.06360*.

Wang, S., J. T. Vogelstein, and C. E. Priebe (2017). Joint embedding of graphs. *arXiv preprint arXiv:1703.03862*.

Wu, Y.-J., E. Levina, and J. Zhu (2017). Generalized linear models with low rank effects for network data. *arXiv preprint arXiv:1705.06772*.

Young, S. J. and E. R. Scheinerman (2007). Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pp. 138–149. Springer.

Zhou, K., L. A. Donnelly, A. D. Morris, P. W. Franks, C. Jennison, C. N. Palmer, and E. R. Pearson (2014). Clinical and genetic determinants of progression of type 2 diabetes: a direct study. *Diabetes care 37*(3), 718–724.

Appendix A

Multiple Random Dot Product Graph

Nielsen, A.M., Witten, D. (2018). *The Multiple Random Dot Product Graph Model*, arXiv preprint <arXiv:1811.12172> (Submitted to *Journal of Computational and Graphical Statistics*)

The Multiple Random Dot Product Graph Model

Agnes M Nielsen
Department of Applied Mathematics and Computer Science,
Technical University of Denmark
and
Daniela Witten *
Departments of Statistics and Biostatistics, University of Washington

July 16, 2018

Abstract

Data in the form of graphs, or networks, arise naturally in a number of contexts; examples include social networks and biological networks. We are often faced with the availability of multiple graphs on a single set of nodes. In this article, we propose the *multiple random dot product graph model* for this setting. Our proposed model leads naturally to an optimization problem, which we solve using an efficient alternating minimization approach. We further use this model as the basis for a new test for the hypothesis that the graphs come from a single distribution, versus the alternative that they are drawn from different distributions. We evaluate the performance of both the fitting algorithm and the hypothesis test in several simulation settings, and demonstrate empirical improvement over existing approaches. We apply these new approaches to a Wikipedia data set and a *C. elegans* data set.

Keywords: embedding, graph inference, hypothesis testing, network data

*D.W. was partially supported by NIH Grants DP5OD009145 and R01GM123993, and NSF CAREER Award DMS-1252624.

1 Introduction

A graph, or a network, consists of a set of vertices, or nodes, and the edges between them. Data in the form of graphs arise in many areas of science. Examples include social networks, communication structures, and biological networks (Pansiot and Grad, 1998; Dawson et al., 2008; Miller et al., 2010).

Let $A \in \{0,1\}^{n \times n}$ denote the $n \times n$ adjacency matrix corresponding to an unweighted and undirected graph with n nodes; $A_{ij} = 1$ if there is an edge between the ith and jth nodes, and $A_{ij} = 0$ otherwise. A number of models for graphs have been proposed in the literature. The simplest is the Erdös-Renyi model (Erdös and Rényi, 1959), in which all edges are assumed independent and equally likely: that is,

$$P(A_{ij} = 1) = \pi,$$

for $\pi \in [0,1]$. The stochastic block model (Holland et al., 1983) generalizes the Erdös-Renyi model by assuming that each node belongs to some latent class, and furthermore that the probability of an edge between a pair of nodes depends on their latent class memberships. Letting $\tau \in \{1, \ldots, M\}^n$ denote the latent class membership vector, and $B \in [0,1]^{M \times M}$ the block connectivity probability matrix, this leads to the model

$$P\left(A|\tau, B\right) = \prod_{i<j} B_{\tau_i,\tau_j}^{A_{ij}} \left(1 - B_{\tau_i,\tau_j}\right)^{(1-A_{ij})}. \tag{1}$$

The stochastic block model is further generalized by latent space models (Hoff et al., 2002; Hoff, 2008, 2009; Ma and Ma, 2017; Wu et al., 2017), which assign each vertex a position in a latent space, and posit that the probability of an edge between two vertices depends on their positions. In particular, in the random dot product graph model (Young and Scheinerman, 2007; Nickel, 2008; Scheinerman and Tucker, 2010), each node is assigned a vector, and the probability of an edge between two nodes is a function of the dot product between the corresponding vectors: that is,

$$P\left(A_{ij} = 1\right) = f\left(x_i^T x_j\right), \tag{2}$$

where $x_1, \ldots, x_n \in \mathbb{R}^d$ are d-dimensional latent vectors associated with the n nodes, and $f(\cdot)$ is a link function. If $f(\cdot)$ is the identity, then this model can be fit via an eigendecomposition. The Erdös-Renyi model is a special case of the random dot product graph, as is the stochastic block model, provided that the matrix B in (1) is positive semi-definite.

The models described above are intended for data that consist of either a single graph, or else multiple graphs that are assumed to be independent and identically distributed draws from a single distribution. However, in many contemporary data settings, researchers collect multiple graphs on a single set of nodes (Ponomarev et al., 2012; Stopczynski et al., 2014; Szklarczyk et al., 2014; Nelson et al., 2017). These graphs may be quite different, and likely are not independent and identically distributed. For instance, if the nodes represent people, then the edges in the two graphs could represent Facebook friendships and Twitter followers, respectively. If the nodes represent proteins, then the edges in the two graphs could represent binary interactions (i.e. a physical contact between a pair of proteins) and co-complex interactions (i.e. whether a pair of proteins are part of the same protein complex), respectively (Yu et al., 2008). Alternatively, if the nodes represent brain regions, then each graph could represent the connectivity among the brain regions for a particular experimental subject (Hermundstad et al., 2013).

Several models have been proposed for this multiple-graph setting (Tang et al., 2009; Shiga and Mamitsuka, 2012; Dong et al., 2014; Durante et al., 2017; Wang et al., 2017). The multiple random eigen graphs (MREG) model (Wang et al., 2017) extends the random dot product graph model: letting $A^1, \ldots, A^K \in \{0,1\}^{n \times n}$ denote K adjacency matrices on a single set of n nodes, this model takes the form

$$P\left(A_{ij}^k = 1\right) = f\left(W_{ij}^k\right), \quad W^k = U\Lambda^k U^T, \quad k = 1, \ldots, K, \tag{3}$$

where U is a $n \times d$ matrix of which the ith row, $u_i = (u_{i1}, \ldots, u_{id})^T \in \mathbb{R}^d$, is the d-dimensional latent vector associated with the ith node across all K graphs; $\Lambda^1, \ldots, \Lambda^K$ are $d \times d$ diagonal matrices; and $f(\cdot) : \mathbb{R} \to [0,1]$ is a link function. If $\Lambda^1 = \ldots = \Lambda^K$ and further $W^1 = \ldots = W^K$ are positive semi-definite, then (3) reduces to the random dot product graph model (2). However, in general, Wang et al. (2017) does not guarantee that $W^1, \ldots, W^K$ be positive definite. In particular, the model (3) is fit using an iterative

approach that estimates one column of U at a time; this algorithm does not guarantee that the columns of U be mutually orthogonal, or that the diagonal elements of $\Lambda^1, \ldots, \Lambda^K$ be nonnegative (Wang et al., 2017).

In this paper, we will consider this multiple graph setting. Our contributions are as follows:

1. We present the *multiple random dot product graph* (multi-RDPG), a refinement of the MREG model (3) of Wang et al. (2017). This model provides a more natural generalization of the random dot product graph (2) to the setting of multiple random graphs, by requiring that the matrices $W^1, \ldots, W^K$ be positive semidefinite. In particular, unlike the proposal of Wang et al. (2017), the multi-RDPG with $K = 1$ reduces to the random dot product graph model.

2. We derive a new algorithm for fitting the multi-RDPG model, which simultaneously estimates all d latent dimensions, while also enforcing their orthogonality. It therefore yields improved empirical results relative to the proposal of Wang et al. (2017).

3. We develop a new approach for testing the hypothesis that multiple graphs are drawn from the same distribution. This approach follows directly from the multi-RDPG model.

The rest of this paper is organized as follows. In Section 2, we present the multi-RDPG model, as well as an algorithm for fitting this model. We develop a test for the null hypothesis that two graphs are drawn from the same RDPG model in Section 3. Results on a Wikipedia data set and a *C. elegans* connectome data set are presented in Sections 4. We close with the Discussion in Section 5.

2 The Multiple Random Dot Product Graph Model

In this section, we will extend the random dot product graph model of Young and Scheinerman (2007) to the setting of multiple unweighted undirected graphs, which we assume to be drawn from related though not necessarily identical distributions.

2.1 The Multiple Random Dot Product Graph Model

To begin, we notice from (2) that in the case of a single unweighted and undirected graph with n nodes, the random dot product graph model assumes that $P(A_{ij} = 1) = f(W_{ij})$, for a rank-$d$ positive semi-definite $n \times n$ matrix $W \equiv U\Lambda U^T$. Here U is an $n \times d$ orthogonal matrix, such that $U^T U = I$, and Λ is a $d \times d$ diagonal matrix with positive elements on the diagonal. The fact that the matrix W is positive semi-definite allows us to interpret the rows of the $n \times d$ matrix $W^{1/2} = U\Lambda^{1/2}$ as the positions of the n nodes in a d-dimensional space, and thus the probability of an edge between a pair of nodes as a function of the distance between the nodes in this d-dimensional space.

To extend this model to the case of K unweighted and undirected graphs, we propose the *multiple random dot product graph* (multi-RDPG) model, which is of the form

$$P\left(A_{ij}^k = 1\right) = f\left(W_{ij}^k\right), \qquad W^k = U\Lambda^k U^T, \qquad k = 1, \ldots, K, \tag{4}$$

where U is an $n \times d$ orthogonal matrix, $\Lambda^1, \ldots, \Lambda^K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements, and $f(\cdot) : \mathbb{R} \to [0, 1]$ is a link function.

While the multi-RDPG (4) appears at first glance quite similar to the MREG formulation (3) of Wang et al. (2017), there are some key differences. In particular, the multi-RDPG model constrains the columns of U to be orthogonal, and the diagonal elements of $\Lambda^1, \ldots, \Lambda^K$ to be nonnegative; by contrast, in (3), these are no such constraints. In greater detail, the differences between the two models are as follows:

1. For $k = 1, \ldots, K$, W^k in (4) is a positive semi-definite matrix of rank d. Thus, the nodes in the kth graph can be viewed as lying in a d-dimensional space, such that the probability of an edge between a pair of nodes is a function of their distance in this space. By contrast, such an interpretation is not possible in the MREG formulation (3), in which there are no guarantees that the matrix W^k is positive semi-definite.

2. With $K = 1$, the multi-RDPG model (4) simplifies to the random dot product graph model (2). The same is not typically true of the MREG model (3), since the matrix W^1 in (3) need not be positive semi-definite.

3. The multi-RDPG model can be efficiently fitted via a single optimization problem, as

detailed in Section 2.2. By contrast, the MREG model (3) is fitted by estimating one dimension at a time (Wang et al., 2017), which can lead to poor results in estimating the entire subspace U in (4).

2.2 Optimization Problem

To derive an optimization problem for fitting the multi-RDPG model (4), we consider the simplest case, in which $f(\cdot)$ is the identity. To motivate our optimization problem, we consider the case of $K = 1$, in which the multi-RDPG model coincides with the random dot product graph model (2). The model (2) is typically fit by solving the optimization problem (Scheinerman and Tucker, 2010)

$$\underset{X \in \mathbb{R}^{n \times d}}{\text{minimize}} \, \|A - XX^T\|_F^2, \tag{5}$$

or a slight modification of (5) if no self-loops are allowed. Let $A = VDV^T$ denote the eigen decomposition of A, where the diagonal elements of D are ordered from largest to smallest. Then, the solution to (5) is $\hat{X} = V_{[1:d]} \left(D^+_{[1:d]} \right)^{1/2}$, where $V_{[1:d]}$ is the $n \times d$ matrix that consists of the first d eigenvectors of A, and where $D^+_{[1:d]}$ is the $d \times d$ diagonal matrix whose diagonal elements are the positive parts of the first d eigenvalues of A. Equivalently, we can fit (2) by solving the problem

$$\underset{U \in \mathbb{R}^{n \times d}, \, U^T U = I, \, \Lambda \in \Delta^d_+}{\text{minimize}} \, \|A_+ - U\Lambda U^T\|_F^2, \tag{6}$$

where $A_+ \equiv VD_+V^T$, $D_+ \equiv D^+_{[1:n]}$, and Δ^d_+ is the set of diagonal $d \times d$ matrices with nonnegative diagonal elements. It is not hard to show that $\hat{X} = \hat{U}\hat{\Lambda}^{1/2}$.

To fit the multi-RDPG model (4), we directly extend the optimization problem (6) to accommodate K adjacency matrices,

$$\underset{U \in \mathbb{R}^{n \times d}, \, U^T U = I, \, \Lambda^1, \ldots, \Lambda^K \in \Delta^d_+}{\text{minimize}} \sum_{k=1}^{K} \|A^k_+ - U\Lambda^k U^T\|_F^2. \tag{7}$$

2.3 Algorithm

While the optimization problem (6) for fitting the random dot product graph model (2) has a closed-form solution, its extension to the multi-RDPG model (7) does not. Thus, to solve (7), we take an alternating minimization approach (Csiszár and Tusnády, 1984), which will rely on the following three results.

First, we derive a majorizing function (Hunter and Lange, 2004) that will prove useful in solving (7).

Proposition 2.1. *The function*

$$g(U) \equiv -2 \sum_{k=1}^{K} \operatorname{trace}\left(U \Lambda^k U^T A_+^k\right)$$

is majorized by

$$h(U \mid U') \equiv -g(U') - 4 \sum_{k=1}^{K} \operatorname{trace}\left(\Lambda^k U'^T A_+^k U\right),$$

in the sense that $g(U) \leq h(U \mid U')$ *and* $g(U') = h(U' \mid U')$.

Next, we use Proposition 2.1 to devise a simple iterative strategy that is guaranteed to decrease the value of the objective of (7) each time U is updated.

Proposition 2.2. *Let* U^{old} *denote an orthogonal* $n \times p$ *matrix, and let* B *and* C *be the matrices whose columns are the left and right singular vectors, respectively, of the matrix* $\sum_{k=1}^{K} A_+^k U^{\text{old}} \Lambda^k$. *Then, for* $U^{\text{new}} \equiv BC^T$,

$$\sum_{k=1}^{K} \|A_+^k - U^{\text{new}} \Lambda^k (U^{\text{new}})^T\|_F^2 \leq \sum_{k=1}^{K} \|A_+^k - U^{\text{old}} \Lambda^k (U^{\text{old}})^T\|_F^2.$$

Finally, we show that with U held fixed, (7) can be solved with respect to $\Lambda^1, \ldots, \Lambda^K$ for the global optimum.

Proposition 2.3. *If* $U^T U = I$, *then the solution to*

$$\underset{\Lambda^1, \ldots, \Lambda^K \in \Delta_+^d}{\text{minimize}} \sum_{k=1}^{K} \|A_+^k - U \Lambda^k U^T\|_F^2 \tag{8}$$

is

$$\Lambda_{jj}^k = \max\left(0, Z_{jj}^k\right), \tag{9}$$

where $Z^k = U^T A_+^k U$.

Propositions 2.1–2.3 immediately suggest an alternating minimization algorithm for solving (7), which is summarized in Algorithm 1.

Algorithm 1 Alternating Minimization Algorithm for Solving (7)

1: For $k = 1, \ldots, K$, initialize Λ^k to be a $d \times d$ diagonal matrix with nonnegative diagonal elements.
2: Initialize U^{old} to be an orthogonal $n \times d$ matrix.
3: For $k = 1, \ldots, K$, let VDV^T denote the eigendecomposition of A^k, and define $A_+^k \equiv VD_+V^T$, where D_+ is the diagonal matrix with diagonal elements $(D_+)_{ii} = \max(D_{ii}, 0)$ for $i = 1, \ldots, n$.
4: **while** not converged **do**
5: Define the matrices B and C to have as their columns the left and right singular vectors, respectively, of the matrix $\sum_{k=1}^K A_+^k U^{\text{old}} \Lambda^k$. Then, update $U \leftarrow BC^T$.
6: For $k = 1, \ldots, K$ and $j = 1, \ldots, n$, update $\Lambda_{jj}^k \leftarrow \max(0, Z_{jj})$, where Z_{jj} is the jth diagonal element of the matrix $Z = U^T A_+^k U$.
7: Update $U^{\text{old}} \leftarrow U$.
8: **end while**

2.4 Simulation Study

We conducted two simulation studies in order to evaluate the performance of Algorithm 1 for fitting the multi-RDPG model (4) (multi-RDPG). We compare it to the algorithm of Wang et al. (2017) for fitting the multiple random eigen graph model (3) (MREG) using software obtained from the authors, the random dot product graph (Young and Scheinerman, 2007) fitted to the average of all of the adjacency matrices (RDPG_{all}), and the random dot product graph (Young and Scheinerman, 2007) fitted to the replicates from each model separately ($\text{RDPG}_{\text{separate}}$). The two versions of RDPG are fitted using `R` base functions (R Core Team, 2015).

To quantify the error in estimating the matrix U, we made use of subspace distance (Absil et al., 2006), defined as

$$d_U(\hat{U}, U) \equiv ||P_{\hat{U}} - P_U||_2, \tag{10}$$

where $P_A \equiv A(A^TA)^{-1}A^T$, the notation $\|\cdot\|_2$ indicates the matrix 2-norm, and $\hat{U}$ is an estimate of the matrix U. To quantify the error in estimating $\Lambda^1, \ldots, \Lambda^K$, we computed the adjacency matrix error,

$$d_A(\hat{\Lambda}, \Lambda) \equiv \frac{1}{K}\sum_{k=1}^{K}\left\|f(U\Lambda^k U^T) - \hat{U}\hat{\Lambda}^k\hat{U}^T\right\|_F^2, \tag{11}$$

where $f(\cdot)$ is the link function defined in (4), and where the notation $\|\cdot\|_F$ indicates the Frobenius norm.

2.4.1 Simulation Setting 1

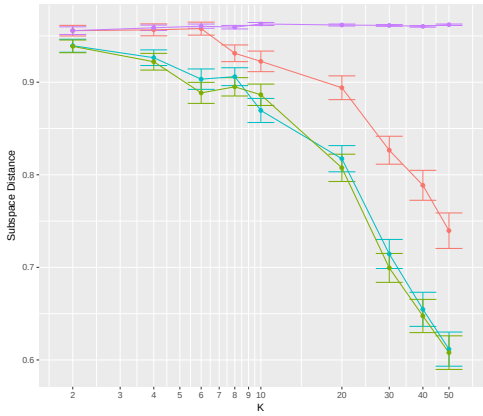
The first simulation setting is based upon the simulation study in Wang et al. (2017). We set $n = 20$ and $d = 3$. We defined $U \in \mathbb{R}^{n\times d}$ to be the orthogonal matrix with columns

$$\begin{aligned}
U_{\cdot 1} &= \begin{pmatrix} 1 & 1 & 1 & 1 & \ldots & 1 \end{pmatrix}^T/\sqrt{n} \\
U_{\cdot 2} &= \begin{pmatrix} 1 & -1 & 1 & -1 & \ldots & 1 & -1 \end{pmatrix}^T/\sqrt{n} \\
U_{\cdot 3} &= \begin{pmatrix} 1 & 1 & -1 & -1 & 1 & 1 & \ldots & -1 & -1 \end{pmatrix}^T/\sqrt{n}.
\end{aligned} \tag{12}$$

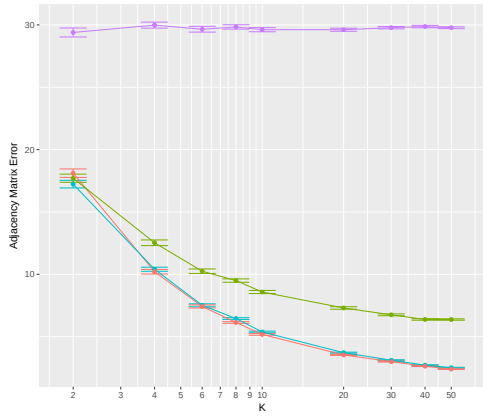
For $k = 1, \ldots, K$, we generated the three diagonal elements of Λ^k independently from uniform distributions: $\Lambda^k_{11} \sim \text{Uniform}(8, 15)$, $\Lambda^k_{22} \sim \text{Uniform}(1, 4)$ and $\Lambda^k_{33} \sim \text{Uniform}(0, 1)$. Note that this choice of parameters results in $U\Lambda^k U^T \in [0, 1]^{n\times n}$. We generated data under the model (4) with f as the identity, so that $P\left(A^k_{ij} = 1\right) = \left(U\Lambda^k U^T\right)_{ij}$. Under this model, we generated $K \in \{2, 4, 6, 8, 10, 20, 30, 40, 50\}$ graphs. We obtained $\text{RDPG}_{\text{separate}}$ by fitting a separate RDPG model to each adjacency matrix, and we obtained RDPG_{all} by fitting a single RDPG model to all K adjacency matrices.

Results, averaged over 100 simulated data sets, are shown in Figure 1. Figure 1(a) shows the subspace distance defined in (10), and Figure 1(b) shows the adjacency matrix error defined in (11). RDPG_{all} performs the best in terms of subspace distance because it pools adjacency matrices that share the same eigenvectors, but very poorly in terms of adjacency matrix error since it erroneously assumes that $\Lambda^k = \Lambda^{k'}$ for all $k \neq k'$. $\text{RDPG}_{\text{separate}}$ performs poorly using both subspace distance and adjacency error, because

it fails to share information across the adjacency matrices. Multi-RDPG (Algorithm 1) performs well across the board, and in particular outperforms MREG (Wang et al., 2017) in terms of recovering the matrix U.



(a) Subspace distance, d_U (10), between the true U and estimated $\hat{U}$.



(b) Adjacency matrix error, d_Λ (11), between the true $\Lambda^1, \ldots, \Lambda^K$ and estimated $\hat{\Lambda}^1, \ldots, \hat{\Lambda}^K$.

Figure 1: Results of multi-RDPG (Algorithm 1) (blue), MREG (Wang et al., 2017) (red), RDPG_{all} (green), and $\text{RDPG}_{\text{separate}}$ (purple), in Simulation Setting 1. The figures display the mean and standard error, over 100 simulated data sets, of the distance measures defined in (10) and (11).

2.4.2 Simulation Setting 2

Next, we expanded Simulation Setting 1 in order to investigate the performance of our multi-RDPG proposal (Algorithm 1) in a setting where the diagonal elements of Λ^k are in different orders for $k = 1, \ldots, K$. Once again, we let $n = 20$ and $d = 3$, with U defined in (12). For k an even number, we set $\Lambda^k = \Lambda^{\text{even}} \equiv \text{diag}(11.5, 2, 0.5)$. For k an odd number, we set $\Lambda^k = \Lambda^{\text{odd}}$, a diagonal matrix for which the diagonal elements are one of six possible permutations of the numbers 11.5, 2, and 0.5.

We generated each graph according to,

$$P\left(A_{ij}^k = 1\right) = f\left(\left(U\Lambda^k U^T\right)_{ij}\right), \quad f(x) = \min(\max(0, x), 1). \tag{13}$$

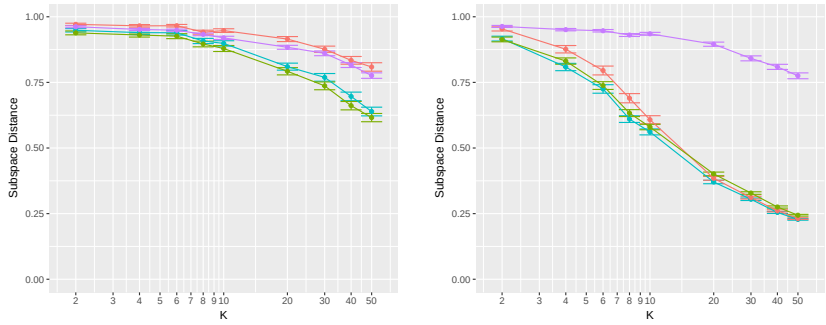
For each of the six permutations leading to Λ^{odd}, we generated $K \in \{2, 4, 6, 8, 10, 20, 30, 40, 50\}$

graphs. Because all of the even-numbered adjacency matrices were drawn from a single generative model, and likewise for the odd-numbered adjacency matrices, we obtained $\text{RDPG}_{\text{separate}}$ by fitting one RDPG to all of the even-numbered adjacency matrices, and a separate RDPG to all of the odd-numbered adjacency matrices.

Results, averaged over 100 simulated data sets, are shown in Figures 2 and 3. Figure 2 shows the subspace distance defined in (10), and Figure 3 shows the adjacency matrix error defined in (11). First, we notice that RDPG_{all} always performs well in terms of subspace distance since it pools adjacency matrices that share the same eigenvectors, but quite poorly in terms of adjacency matrix error when the diagonal elements of Λ^k differ between the evens and the odds (see, for example, panels (c), (d), (e), and (f) of Figure 3). The method $\text{RDPG}_{\text{separate}}$ performs similarly in all setups, since it fits separate models to the even-numbered and odd-numbered adjacency matrices and borrows no strength across them; overall, its performance is quite poor, because it makes use of only half of the available sample size in fitting each RDPG model. We see that multi-RDPG outperforms MREG across the board, using both subspace distance and adjacency matrix error; furthermore, multi-RDPG has the best overall performance.

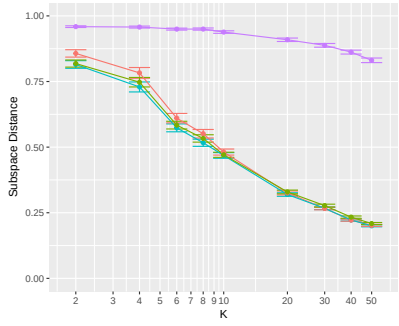
Finally, we notice from Figure 2 that some of the orderings of the eigenvalues in this simulation setting appear to be more challenging than others. For instance, the errors associated with multi-RDPG in panels (b), (c), (e), and (f) of Figure 2 are much lower than those in panels (a) and (d). It turns out that panels (a) and (d) are challenging because the eigenvalue corresponding to the third column of U equals 0.5 in both Λ^{even} and Λ^{odd}, so that the third column of U is hard to recover. By contrast, the setups in Figures (b), (c), (e), and (f) are less challenging because each of the three eigenvectors of U has an eigenvalue that is no smaller than two in either Λ^{even} or Λ^{odd}. Furthermore, the setups in (d) and (e) are additionally challenging because a substantial proportion of the elements of $U\Lambda^{\text{odd}}U^T$ are less than zero or greater than one; these elements are set to zero by $f(\cdot)$ in (13), leading to a substantial loss of information.

To conclude, we find that while RDPG_{all} performs well in terms of subspace distance, it typically performs quite poorly in terms of adjacency matrix error, as expected. $\text{RDPG}_{\text{separate}}$ performs poorly because it only makes use of half of the available adjacency

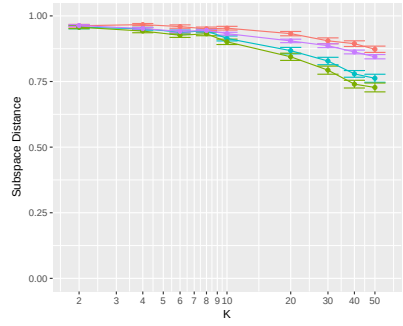


(a) $\Lambda^{\text{odd}} = \text{diag}(11.5, 2, 0.5)$

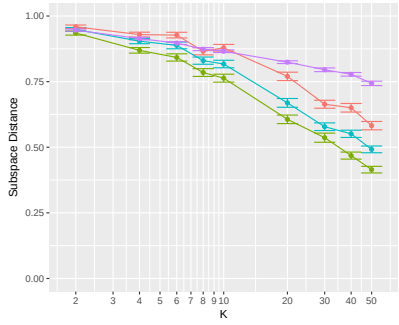
(b) $\Lambda^{\text{odd}} = \text{diag}(11.5, 0.5, 2)$



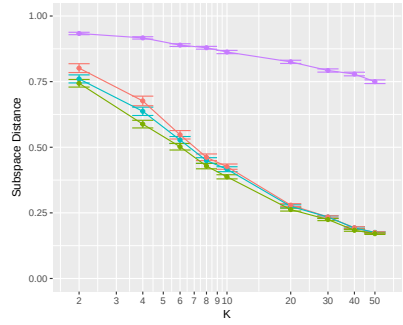
(c) $\Lambda^{\text{odd}} = \text{diag}(2, 0.5, 11.5)$



(d) $\Lambda^{\text{odd}} = \text{diag}(2, 11.5, 0.5)$



(e) $\Lambda^{\text{odd}} = \text{diag}(0.5, 11.5, 2)$



(f) $\Lambda^{\text{odd}} = \text{diag}(0.5, 2, 11.5)$

Figure 2: Results of multi-RDPG (Algorithm 1) (blue), MREG (Wang et al., 2017) (red), RDPG_{all} (green), and $\text{RDPG}_{\text{separate}}$ (purple), in Simulation Setting 2. The figures display the mean and standard error, over 100 simulated data sets, of the subspace distance defined in (10). In each subplot, for k even, $\Lambda^k = \text{diag}(11.5, 2, 0.5)$, and for k odd, Λ^k is as specified in the subplot caption.

matrices. Multi-RDPG has the best overall performance, and substantially outperforms the MREG approach of Wang et al. (2017).

3 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$

In this section, we will develop a test for the null hypothesis that all of the K observed graphs are drawn from the same distribution; this corresponds to the null hypothesis

$$H_0 : \Lambda^1 = \ldots = \Lambda^K \tag{14}$$

in the model (4).

3.1 A Permutation Test

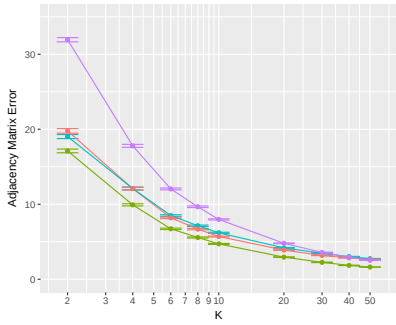
To test $H_0 : \Lambda^1 = \ldots = \Lambda^K$, we take an approach inspired by a likelihood ratio test. The test statistic, $T(A_+^1, \ldots, A_+^K)$, takes the form

$$\begin{aligned} T\left(A_+^1, \ldots, A_+^K\right) = &\left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda \in \Delta_+^d} \sum_{k=1}^K ||A_+^k - U \Lambda U^T||_F^2 \right) \\ &- \left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda^1, \ldots, \Lambda^K \in \Delta_+^d} \sum_{k=1}^K ||A_+^k - U \Lambda^k U^T||_F^2 \right). \end{aligned} \tag{15}$$

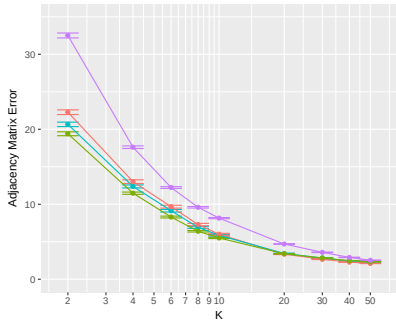
(Recall that A_+^k was defined in Section 2.2.) Computing the second term in $T(A_+^1, \ldots, A_+^K)$ is straightforward, using Algorithm 1. To compute the first term, we will make use of the following result.

Proposition 3.1. *Let QDQ^T denote the eigen decomposition of $\frac{1}{K}\sum_{k=1}^K A_+^k$, where the diagonal elements of $D = \mathrm{diag}(\alpha_1, \ldots, \alpha_n)$ are in non-increasing order, i.e. $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n$. Then, the solution to*

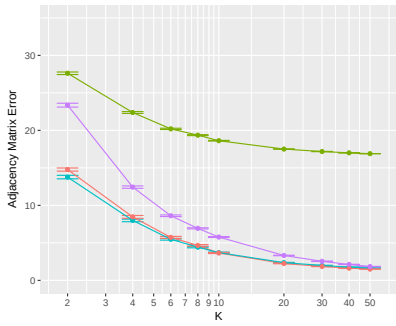
$$\underset{U \in \mathbb{R}^{n \times d}, U^T U = I, \Lambda \in \Delta_+^d}{\text{minimize}} \left\{ \sum_{k=1}^K ||A_+^k - U \Lambda U^T||_F^2 \right\} \tag{16}$$



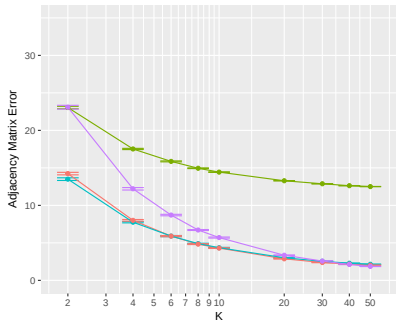
(a) $\Lambda^{\text{odd}} = \text{diag}(11.5, 2, 0.5)$



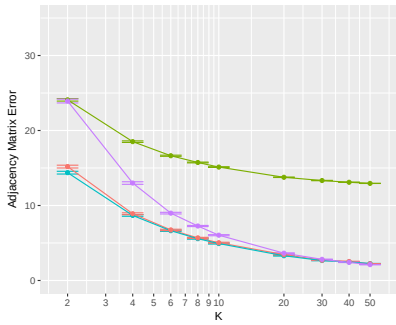
(b) $\Lambda^{\text{odd}} = \text{diag}(11.5, 0.5, 2)$



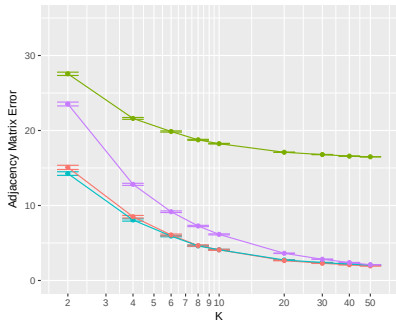
(c) $\Lambda^{\text{odd}} = \text{diag}(2, 0.5, 11.5)$



(d) $\Lambda^{\text{odd}} = \text{diag}(2, 11.5, 0.5)$



(e) $\Lambda^{\text{odd}} = \text{diag}(0.5, 11.5, 2)$



(f) $\Lambda^{\text{odd}} = \text{diag}(0.5, 2, 11.5)$

Figure 3: Results of multi-RDPG (Algorithm 1) (blue), MREG (Wang et al., 2017) (red), RDPG_{all} (green), and $\text{RDPG}_{\text{separate}}$ (purple), in Simulation Setting 2. The figures display the mean and standard error, over 100 simulated data sets, of the adjacency matrix error defined in (11). In each subplot, for k even, $\Lambda^k = \text{diag}(11.5, 2, 0.5)$, and for k odd, Λ^k is as specified in the subplot caption.

is that the d columns of U are the first d columns of Q, and the diagonal elements of Λ are $\alpha_1, \ldots, \alpha_d$.

The proof is provided in Appendix A.

We compute a p-value for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ by comparing the magnitude of $T(A_+^1, \ldots, A_+^K)$ to its null distribution, obtained by permutations. Details are provided in Algorithm 2.

Algorithm 2 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (4)

1. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (15).
2. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{1,*b}\right)_{ij}, \ldots, \left(A^{K,*b}\right)_{ij}$ be a random permutation of $\left(A^1\right)_{ij}, \ldots, \left(A^K\right)_{ij}$.
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (15).
3. Compute the p-value,

$$p = \frac{1}{B} \sum_{b=1}^{B} I_{\left\{T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right) \geq T\left(A_+^1, \ldots, A_+^K\right)\right\}},$$

where I_C is an indicator function that equals one if the event C holds, and equals zero otherwise.

3.2 Simulation Study

We now conduct a simulation study to demonstrate that the p-values for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ obtained via Algorithm 2 are uniformly distributed under the null hypothesis, and that the test has power under the alternative.

3.2.1 Type I Error Control

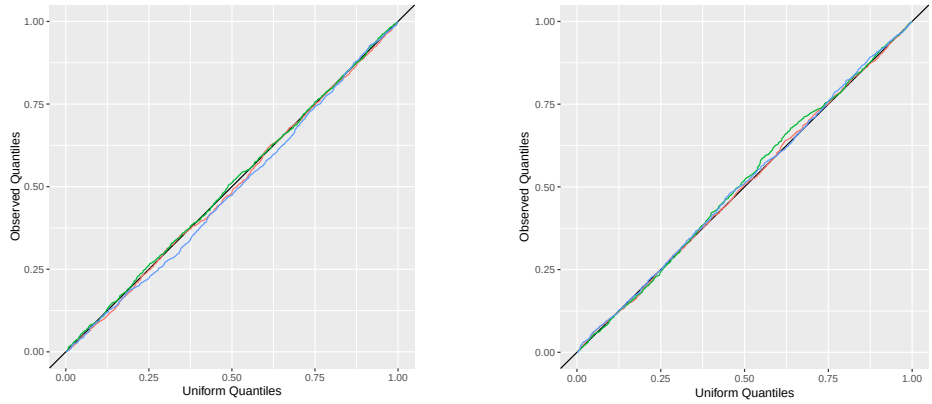
To explore the Type I error of the test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ proposed in Algorithm 2, we generate $K = 2$ graphs with

$$U = [[1, \ldots, 1]^T, [1, -1, 1, -1, \ldots, -1]^T]/\sqrt{n}, \tag{17}$$

$\Lambda^1 = \Lambda^2 = \text{diag}(n/4, n/5)$, for $n \in \{20, 50, 100\}$. For $k = 1, 2$, we then generated the adjacency matrix A^k according to (4), with $f(\cdot)$ the identity. We then computed a p-value according to Algorithm 2. We simulated data in this way 1,000 times, and obtained 1,000 p-values, shown in Figure 4(a).

We also repeated this procedure using $\Lambda^1 = \Lambda^2 = \text{diag}(n/2, n/4, n/400)$ and U defined as in (12). The p-values are shown in Figure 4(b).

In both panels of Figure 4, we see that the p-values are uniformly distributed, indicating adequate control of Type I error.



(a) $\Lambda = \text{diag}(n/4, n/5)$ (b) $\Lambda = \text{diag}(n/2, n/4, n/400)$

Figure 4: Quantile-quantile plots of p-values from Algorithm 2 against a uniform distribution. The colors indicate the value of n: $n = 20$ (red), $n = 50$ (green), and $n = 100$ (blue). *(a):* Data were generated according to (4) with $\Lambda^1 = \Lambda^2 = \text{diag}(n/4, n/5)$ and U as in (17). *(b):* Data were generated according to (4) with $\Lambda^1 = \Lambda^2 = \text{diag}(n/2, n/4, n/400)$ and U as in (12).

3.2.2 Power

Next, in order to explore the power of the test described in Algorithm 2, we generated data for which $H_0 : \Lambda^1 = \ldots = \Lambda^K$ does not hold.

First, for $r \in [0, 1]$, and for $n \in \{20, 50, 100\}$, we generated Λ^1 and Λ^2 as follows:

$$\Lambda^1 = \text{diag}\left(\frac{n}{4}(1+r), \frac{n}{4}(1-r)\right), \quad \Lambda^2 = \text{diag}\left(\frac{n}{4}(1-r), \frac{n}{4}(1+r)\right). \tag{18}$$

We generated U as in (17). We then generated A^1 and A^2 according to (4), with $f(u) = \max(u, 0)$. A p-value for $H_0 : \Lambda^1 = \Lambda^2$ was then obtained using Algorithm 2. This was repeated 1,000 times, leading to p-values $p_1, \ldots, p_{1000}$. The power at level $\alpha = 0.05$, computed as

$$\text{power} = \frac{1}{1000}\sum_{i=1}^{1000} I_{(p_i<\alpha)}, \tag{19}$$

where I_C is an indicator function for the event C, is displayed in Figure 5(a). As expected, the power increases as r increases, and as n increases.

Next, we defined U as in equation (12) with

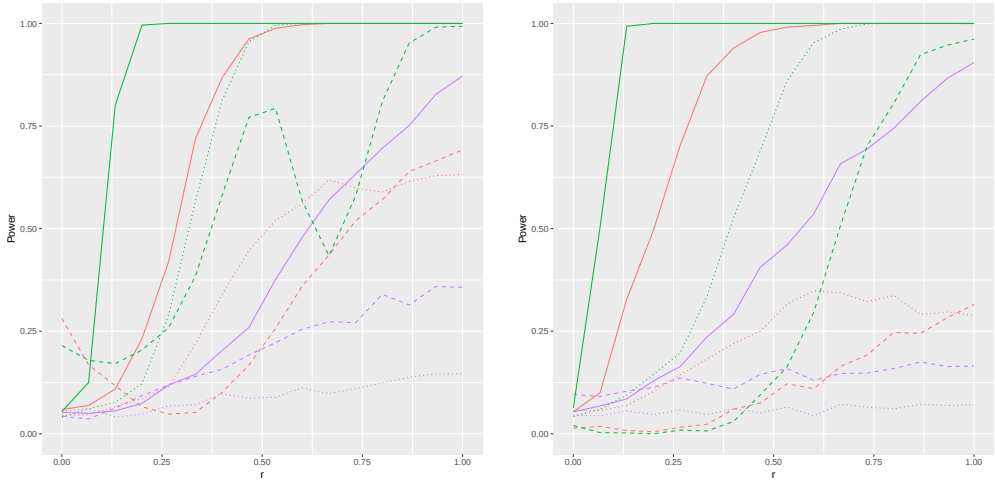
$$\begin{aligned}\Lambda^1 &= \text{diag}\left(\frac{n}{4}(1-r), \frac{n}{5}(1+r), \frac{n}{400}(1-r)\right), \\ \Lambda^2 &= \text{diag}\left(\frac{n}{4}(1+r), \frac{n}{5}(1-r), \frac{n}{400}(1+r)\right).\end{aligned} \tag{20}$$

The resulting power is shown in Figure 5(b); we see once again that power increases as r and n increase.

We now compare our test to a test proposed by Tang et al. (2017). Under the assumption that both graphs are drawn from a RDPG model (2), Tang et al. (2017) fits the RDPG to each graph, in order to obtain $\hat{X}^1 \equiv \begin{pmatrix}\hat{x}_1^1 & \ldots & \hat{x}_n^1\end{pmatrix}^T$ and $\hat{X}^2 \equiv \begin{pmatrix}\hat{x}_1^2 & \ldots & \hat{x}_n^2\end{pmatrix}^T$, the two sets of estimated latent vectors. They then calculate the test statistic $T = \min_W \|\hat{X}^1 - \hat{X}^2 W\|_F$, and also calculate the values of this test statistic under a null distribution obtained via the parametric bootstrap. This is then used to obtain a p-value for the null hypothesis that the two graphs are drawn from the same RDPG model. We fit the model using software obtained from the authors of Tang et al. (2017).

We see from Figure 5 that with data generated under (18) and (20), our test has higher

power than that of Tang et al. (2017). Under (18), the proposal of Tang et al. (2017) fails to control Type I error: it rejects the null hypothesis in approximately 20% of simulated data sets with $n = 50$ for which $H_0 : \Lambda^1 = \Lambda^2$ holds. A slight modification to their algorithm that replaces the parametric bootstrap with the non-parametric swapping approach used in Algorithm 2 leads to proper Type I error control, but lower power than our approach.



(a) Λ^1, Λ^2 in (18), and U in (17)

(b) Λ^1, Λ^2 in (20), and U in (12)

Figure 5: The power of the test in Algorithm 2 (solid), along with the powers of the test of Tang et al. (2017) (dash) and the test of Tang et al. (2017) with a modification to the bootstrap method (dotted). Colors indicate the value of n: $n = 10$ (purple), $n = 20$ (red), and $n = 50$ (green). *(a)*: Λ^1 and Λ^2 are defined in (18), and U is defined in (17). *(b)*: Λ^1 and Λ^2 are defined in (20), and U is defined in (12).

4 Application to Data

4.1 Wikipedia data

To begin, we consider graphs representing a subset of Wikipedia (Suwan et al., 2016). The data set is accessible at `http://www.cis.jhu.edu/~parky/Data/data.html`. The 1,383 vertices represent Wikipedia pages that are available in both French and English. An English graph is constructed by placing an edge between a pair of vertices if the English

version of either page hyperlinks to the other; the French graph is constructed analogously. We wish to test whether the two graphs are drawn from the same distribution, i.e. we wish to test the null hypothesis $H_0 : \Lambda^{\text{English}} = \Lambda^{\text{French}}$ in the model (4).

The English graph contains 18,857 edges, whereas the French graph contains only 14,973 edges. Therefore, we randomly down-sample the edges in the English graph so that both graphs contain 14,973 edges. We then test $H_0 : \Lambda^{\text{English}} = \Lambda^{\text{French}}$ using Algorithm 2. This results in a test statistic value of $T(A_+^{\text{English}}, A_+^{\text{French}}) = 93.08$ and a p-value of $p = 0$. Therefore, we reject the null hypothesis that the two graphs come from the same distribution.

As a point of comparison, we also construct two graphs by randomly sampling 15,000 edges from the English graph twice. This gives two largely overlapping graphs. We test whether these graphs are drawn from the same distribution using Algorithm 2. This results in a test statistic value of $T(A_+^{\text{Sample 1}}, A_+^{\text{Sample 2}}) = 0.08$, and a p-value of $p = 0.81$. Therefore, as expected, we fail to reject the null hypothesis that the two graphs come from the same distribution.

4.2 *C. elegans* Data

We now consider brain networks in *C. elegans*, a small transparent roundworm (Varshney et al., 2011; Chen et al., 2016). The data set is accessible at `https://neurodata.io/project/connectomes/`. The 253 vertices represent neurons. We consider two graphs: a chemical graph, in which two vertices are connected by an edge if there is a chemical synapse between them, and an electrical graph, in which two vertices are connected by an edge if there is an electrical junction potential between them. We wish to test whether the chemical graph and the electrical graph come from the same distribution. Because the chemical graph has 1695 edges and the electrical graph has only 517 edges, we randomly down-sampled the edges in the former to obtain a graph with only 517 edges.

We tested the null hypothesis $H_0 : \Lambda^{\text{Chemical}} = \Lambda^{\text{Electrical}}$ using Algorithm 2. This yields a test statistic of $T(A_+^{\text{Electrical}}, A_+^{\text{Chemical}}) = 18.06$ and a p-value of 0. This leads us to reject the null hypothesis.

5 Discussion

In this paper, we propose the multi-RDPG model, a direct extension of the RDPG model to the setting of multiple graphs. This new model is closely related to the MREG model of Wang et al. (2017). Unlike the MREG model, the multi-RDPG requires the latent vectors to be orthogonal and the corresponding values to be non-negative, so that the multi-RDPG is a direct generalization of the RDPG. Furthermore, we propose a procedure for fitting the multi-RDPG model that allows us to estimate all latent vectors simultaneously, leading to improved empirical results. Finally, we present an approach for testing whether the eigenvalues are equal across the graphs, which controls type I error and has adequate power against the alternative.

In this paper, we have taken the link function $f(\cdot)$ in (4) to be the identity, in the interest of simplicity. However, it would be more natural to choose $f(\cdot)$ to be a function that maps from $\mathbb{R}$ to $[0, 1]$, such as the logistic function, $f(x) = \exp(x)/(1 + \exp(x))$. This would require only a modest modification to Algorithm 1. We leave the details to future work.

The R package `multiRDPG` will be posted on CRAN.

Acknowledgments

We thank Shangsi Wang and Minh Tang at Johns Hopkins University for providing software implementing the methods in Wang et al. (2017) and Tang et al. (2017), respectively. We thank Debarghya Ghoshdastidar at Eberhard Karls University of Tübingen for helpful answers to our questions. D.W. was partially supported by NIH Grants DP5OD009145 and R01GM123993, and NSF CAREER Award DMS-1252624.

A Proofs

Proof of Proposition 2.1.

Proof. Lieb's Concavity Theorem (Lieb, 1973) states that $\text{trace}(K^T L^p K B^r)$ is convex in K if L and B are positive semi-definite and $p, r \geq 0$ and $p + r \leq 1$. Applying Lieb's

Concavity Theorem with $p = r = 0.5$, $K = U^T$, $L = \left(\Lambda^k\right)^2$, and $B = \left(A_+^k\right)^2$, we see that trace $\left(U\Lambda^k U^T A_+^k\right)$ is convex in U. Therefore, $g(U) \equiv -2\sum_{k=1}^K \text{trace}\left(U\Lambda^k U^T A_+^k\right)$ is concave in U.

By definition of concavity (Boyd and Vandenberghe, 2004),

$$g(U) \leq g(U') + \text{trace}\left(\left(\nabla g(U')\right)^T (U - U')\right).$$

Recalling that $\frac{\partial \text{trace}(XBX^TC)}{\partial X} = C^T X B^T + CXB$, it follows that

$$\nabla g(U) = -4\sum_{k=1}^K A_+^k U \Lambda^k.$$

Therefore,

$$\begin{aligned} g(U) &\leq g(U') - 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k (U - U')\right) \\ &= g(U') - 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k U\right) + 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k U'\right) \\ &= -g(U') - 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k U\right). \end{aligned}$$

Therefore, we have shown that $h(U|U') = -g(U') - 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k U\right)$ majorizes $g(U)$. □

Proof of Proposition 2.2.

Proof. We can see by inspection that in order to minimize

$$\sum_{k=1}^K \|A_+^k - U\Lambda^k U^T\|_F^2 \tag{21}$$

with respect to an orthogonal matrix U, it suffices to minimize $g(U)$ in Proposition 2.1. Recall from that proposition that $g(U) \leq h(U \mid U') = -g(U') - 4\sum_{k=1}^K \text{trace}\left(\Lambda^k U'^T A_+^k U\right)$. Taking a majorization-minimization approach (Hunter and Lange, 2004), we can decrease

the objective of (21) evaluated at U' by solving the optimization problem

$$\underset{U\in\mathbb{R}^{n\times d},\, U^TU=I}{\text{minimize}} \left\{ -g(U') - 4\sum_{k=1}^{K} \text{trace}\left(\Lambda^k U'^T A_+^k U\right) \right\},$$

where U' is the value of U obtained in the previous iteration. This is equivalent to solving

$$\underset{U\in\mathbb{R}^{n\times d},\, U^TU=I}{\text{minimize}} \left\{ -\text{trace}\left(\left(\sum_{k=1}^{K} \Lambda^k U'^T A_+^k \right) U \right) \right\},$$

or equivalently, to solving

$$\underset{U\in\mathbb{R}^{n\times d},\, U^TU=I}{\text{minimize}} \left\{ \left\| \sum_{k=1}^{K} A_+^k U' \Lambda^k - U \right\|_F^2 \right\}.$$

This is an orthogonal Procrustes problem (Schönemann, 1966), for which the solution is BC^T, where the columns of B and C are the left and right singular vectors, respectively, of the matrix $\sum_{k=1}^{K} A_+^k U' \Lambda^k$. □

Proof of Proposition 2.3

Proof. We see by inspection that in order to solve (8), it suffices to solve

$$\underset{\Lambda^1,\ldots,\Lambda^K\in\Delta_+^d}{\text{minimize}} \left\{ \sum_{k=1}^{K} \left(-2\text{trace}\left(U^T A_+^k U \Lambda^k\right) + \sum_{j=1}^{n} \left(\Lambda_{jj}^k\right)^2 \right) \right\},$$

or equivalently, to solve

$$\underset{\Lambda_{jj}^k\geq 0}{\text{minimize}} \left\{ -2Z_{jj}^k \Lambda_{jj}^k + \left(\Lambda_{jj}^k\right)^2 \right\}$$

for $j = 1,\ldots,n$ and $k = 1,\ldots,K$, for $Z^k = U^T A_+^k U$. The result follows directly. □

Proof of Proposition 3.1

Proof. Because U is orthogonal and Λ is diagonal, we have that

$$\|A_+^k - U\Lambda U^T\|_F^2 = \|A_+^k\|_F^2 - 2\text{trace}(U\Lambda U^T A_+^k) + \sum_{j=1}^{d} \Lambda_{jj}^2, \tag{22}$$

where Λ_{jj} is the jth diagonal element of Λ. Thus, the optimization problem in (16) can be re-written as

$$\underset{U\in\mathbb{R}^{n\times d},\, U^TU=I,\, \Lambda\in\Delta_+^d}{\text{minimize}} \left\{ \left\| \frac{1}{K}\sum_{k=1}^{K} A_+^k - U\Lambda U^T \right\|_F^2 \right\}. \tag{23}$$

Notice that $\frac{1}{K}\sum_{k=1}^{K} A_+^k$ is the sum of positive semidefinite matrices, and is therefore positive semidefinite. The result follows directly from the fact that the truncated singular value decomposition yields the best approximation to a matrix in terms of Frobenius error (Eckart and Young, 1936; Johnson, 1963). □

References

Abbe, E., A. S. Bandeira, and G. Hall (2016). Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory 62*(1), 471–487.

Absil, P.-A., A. Edelman, and P. Koev (2006). On the largest principal angle between random subspaces. *Linear Algebra and Its Applications 414*(1), 288–294.

Boyd, S. and L. Vandenberghe (2004). *Convex Optimization*. Cambridge University Press.

Celisse, A., J.-J. Daudin, L. Pierre, et al. (2012). Consistency of maximum-likelihood and variational estimators in the stochastic block model. *Electronic Journal of Statistics 6*, 1847–1899.

Chen, L., J. T. Vogelstein, V. Lyzinski, and C. E. Priebe (2016). A joint graph inference case study: the c. elegans chemical and electrical connectomes. *Worm 5*(2), e1142041.

Csiszár, I. and G. Tusnády (1984). Information geometry and alternating minimization procedures. *Statistics and Decisions 1*, 205–237.

Dawson, S. et al. (2008). A study of the relationship between student social networks and sense of community. *Educational Technology & Society 11*(3), 224–238.

Decelle, A., F. Krzakala, C. Moore, and L. Zdeborová (2011). Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E 84*(6), 066106.

Dong, X., P. Frossard, P. Vandergheynst, and N. Nefedov (2014). Clustering on multi-layer graphs via subspace analysis on grassmann manifolds. *IEEE Transactions on Signal Processing 62*(4), 905–918.

Durante, D., D. B. Dunson, and J. T. Vogelstein (2017). Nonparametric Bayes modeling of populations of networks. *Journal of the American Statistical Association 112*(520), 1516–1530.

Eckart, C. and G. Young (1936). The approximation of one matrix by another of lower rank. *Psychometrika 1*(3), 211–218.

Erdös, P. and A. Rényi (1959). On random graphs, i. *Publicationes Mathematicae (Debrecen) 6*, 290–297.

Hermundstad, A. M., D. S. Bassett, K. S. Brown, E. M. Aminoff, D. Clewett, S. Freeman, A. Frithsen, A. Johnson, C. M. Tipper, M. B. Miller, et al. (2013). Structural foundations of resting-state and task-based functional connectivity in the human brain. *Proceedings of the National Academy of Sciences 110*(15), 6169–6174.

Hoff, P. (2008). Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, pp. 657–664.

Hoff, P. D. (2009). Multiplicative latent factor models for description and prediction of social networks. *Computational and Mathematical Organization Theory 15*(4), 261.

Hoff, P. D., A. E. Raftery, and M. S. Handcock (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association 97*(460), 1090–1098.

Holland, P. W., K. B. Laskey, and S. Leinhardt (1983). Stochastic blockmodels: First steps. *Social Networks 5*(2), 109–137.

Hunter, D. R. and K. Lange (2004). A tutorial on MM algorithms. *The American Statistician 58*(1), 30–37.

Johnson, R. M. (1963). On a theorem stated by eckart and young. *Psychometrika 28*(3), 259–263.

Lieb, E. H. (1973). Convex trace functions and the Wigner-Yanase-Dyson conjecture. *Advances in Mathematics 11*(3), 267–288.

Ma, Z. and Z. Ma (2017). Exploration of large networks via fast and universal latent space model fitting. *arXiv preprint arXiv:1705.02372*.

Miller, J. A., S. Horvath, and D. H. Geschwind (2010). Divergence of human and mouse brain transcriptome highlights Alzheimer disease pathways. *Proceedings of the National Academy of Sciences 107*(28), 12698–12703.

Nelson, B. G., D. S. Bassett, J. Camchong, E. T. Bullmore, and K. O. Lim (2017). Comparison of large-scale human brain functional and anatomical networks in schizophrenia. *NeuroImage: Clinical 15*, 439–448.

Nickel, C. L. M. (2008). *Random Dot Product Graphs A Model For Social Networks*. Ph. D. thesis, The Johns Hopkins University.

Pansiot, J.-J. and D. Grad (1998). On routes and multicast trees in the internet. *ACM SIGCOMM Computer Communication Review 28*(1), 41–50.

Ponomarev, I., S. Wang, L. Zhang, R. A. Harris, and R. D. Mayfield (2012). Gene coexpression networks in human brain identify epigenetic modifications in alcohol dependence. *Journal of Neuroscience 32*(5), 1884–1897.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Scheinerman, E. R. and K. Tucker (2010). Modeling graphs using dot product representations. *Computational Statistics 25*(1), 1–16.

Schönemann, P. H. (1966). A generalized solution of the orthogonal Procrustes problem. *Psychometrika 31*(1), 1–10.

Shiga, M. and H. Mamitsuka (2012). A variational Bayesian framework for clustering with multiple graphs. *IEEE Transactions on Knowledge and Data Engineering 24*(4), 577–590.

Stopczynski, A., V. Sekara, P. Sapiezynski, A. Cuttone, M. M. Madsen, J. E. Larsen, and S. Lehmann (2014). Measuring large-scale social networks with high resolution. *PloS One 9*(4), e95978.

Suwan, S., D. S. Lee, R. Tang, D. L. Sussman, M. Tang, C. E. Priebe, et al. (2016). Empirical Bayes estimation for the stochastic blockmodel. *Electronic Journal of Statistics 10*(1), 761–782.

Szklarczyk, D., A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K. P. Tsafou, et al. (2014). String v10: Protein–protein interaction networks, integrated over the tree of life. *Nucleic Acids Research 43*(D1), D447–D452.

Tang, M., A. Athreya, D. L. Sussman, V. Lyzinski, Y. Park, and C. E. Priebe (2017). A semiparametric two-sample hypothesis testing problem for random graphs. *Journal of Computational and Graphical Statistics 26*(2), 344–354.

Tang, W., Z. Lu, and I. S. Dhillon (2009). Clustering with multiple graphs. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pp. 1016–1021. IEEE.

Varshney, L. R., B. L. Chen, E. Paniagua, D. H. Hall, and D. B. Chklovskii (2011). Structural properties of the caenorhabditis elegans neuronal network. *PLoS Comput Biol 7*(2), e1001066.

Wang, L., Z. Zhang, and D. Dunson (2017). Common and individual structure of multiple networks. *arXiv preprint arXiv:1707.06360*.

Wang, S., J. T. Vogelstein, and C. E. Priebe (2017). Joint embedding of graphs. *arXiv preprint arXiv:1703.03862*.

Wu, Y.-J., E. Levina, and J. Zhu (2017). Generalized linear models with low rank effects for network data. *arXiv preprint arXiv:1705.06772*.

Young, S. J. and E. R. Scheinerman (2007). Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pp. 138–149. Springer.

Yu, H., P. Braun, M. A. Yıldırım, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis, et al. (2008). High-quality binary protein interaction map of the yeast interactome network. *Science 322*(5898), 104–110.

Appendix B

Generalized Multiple Random Dot Product Graph

Nielsen, A. M., Svendsen, K. D., Clemmensen, L. (2018) *Hypothesis Testing in the Generalized Multiple Random Dot Product Graph Model*, (draft intended for a statistics journal)

Hypothesis Testing in the Generalized Multiple Random Dot Product Graph Model

Agnes M Nielsen, Kira Dynnes Svendsen, Line Clemmensen
Department of Applied Mathematics and Computer Science,

Technical University of Denmark

December 30, 2018

Abstract

Weighted network (or graph) data arises in many contexts. In this paper, we consider the setting of multiple weighted graphs on the same set of nodes. We present a generalization of the multiple random dot product graph model for weighted graphs. We focus on Poisson and normally distributed edge weights when fitting the model. We further present a test for the hypothesis that the graphs are drawn from a single distribution in these two cases. The performance of the tests are evaluated in several simulation settings. The tests with Poisson and normally distributed edge weights are applied to a Wikipedia and a Oribatid mites data set, respectively.

Keywords: embedding, graph inference, hypothesis testing, weighted network data

1 Introduction

Let A denote the $n \times n$ adjacency matrix corresponding to an undirected graph with n nodes where $A_{ij} \in \mathbb{R}$. As the graph is undirected A is symmetric. If the graph is unweighted then $A_{ij} \in \{0, 1\}$ with $A_{ij} = 1$ if there is an edge between the ith and jth nodes and $A_{ij} = 0$ otherwise. A vast number of random graph models exists for the unweighted graphs in literature. Mentioning a few, there is the simple Erdös-Rényi model where all edges are assumed independent and equally likely (Erdös and Rényi, 1959). There is the stochastic block model where each node belongs to a class and the edge probabilities are dependent on the class memberships (Holland et al., 1983). Then there are the latent space models (Hoff et al., 2002; Hoff, 2008, 2009; Ma and Ma, 2017; Wu et al., 2017) and in particular the random dot product graph model (Young and Scheinerman, 2007; Nickel, 2008; Scheinerman and Tucker, 2010). In this model, each node is assigned a vector and the probability of an edge between two nodes is a function of the dot product between the associated vectors:

$$P(A_{ij} = 1) = f(x_i^T x_j), \tag{1}$$

where $x_i \in \mathbb{R}^d$ is the d-dimensional vector associated with the ith node and $f : \mathbb{R} \to [0, 1]$ is a link function.

In the case of a weighted graph the above models are not appropriate, but generalized versions exist. Erdös-Rényi is generalized by Garlaschelli (2009), the stochastic block model is generalized by Aicher et al. (2013) and the random dot product graph has been extended to weighted graphs by DeFord and Rockmore (2016), and by Tang (2017) who reaches a similar definition.

The weighted RDPG by DeFord and Rockmore (2016) is defined as follows: Let $\nu(\theta)$ be a parametric probability distribution with L parameters $\theta = (\theta^1, \ldots, \theta^L)$. Given the number of nodes n in the graph and dimensions d^l for each parameter θ^l, the weighted random dot product graph model is:

$$A_{ij} \sim \nu(\theta), \quad \theta^l = (x_i^l)^T x_j^l \tag{2}$$

where $A_{ji} = A_{ij}\ \forall (i, j) \in \{1, \ldots, n\}^2$ and $x_1^l, \ldots, x_n^l \in \mathbb{R}^d$ for $l = 1, \ldots, L$ are d-dimensional

vectors associated with the n nodes. This definition only considers the identity link function. DeFord and Rockmore (2016) consider the Poisson distribution when estimating the parameters due to simplicity. However, the Poisson distribution is natural to consider as many weighted graphs arise as "multigraphs", i.e. graphs where multiple edges are allowed between two nodes. These graphs can be seen as weighted graphs where the edges weights are non-negative integers corresponding to the multiplicity of the edge. We will consider both the Poisson and normal distributions for edge weights. The latter often arises in practical examples and is a natural next step but is more complicated as it has two parameters.

The models described above are of single or multiple graphs which are assumed independent and identically distributed. However, turning out attention to a situation where we consider multiple graphs on the same set of nodes, these graphs are unlikely to be independent and identically distributed. Several models exist for this setting (Tang et al., 2009; Shiga and Mamitsuka, 2012; Dong et al., 2014; Durante et al., 2017; Wang et al., 2017; Nielsen and Witten, 2018).

The multiple random dot product graph (multi-RDPG) for a collection unweighted graphs on the same set of nodes is an extension of the random dot product graph (Nielsen and Witten, 2018). That is letting $A^1, \dots, A^K \in \{0,1\}^{n\times n}$ be the adjacency matrices for K graphs on the same set of nodes,

$$P(A^k_{ij} = 1) = f(W^k_{ij}), \quad W^k = U\Lambda^k U^T, \quad k = 1, \dots, K. \tag{3}$$

where U is an $n \times d$ orthogonal matrix and $\Lambda^1, \dots, \Lambda^K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements, and $f : \mathbb{R} \to [0,1]$ is a link function.

In this article, we extend the multiple random dot product graph to weighted graphs. Our contributions are as follows:

1. We present the generalized multiple random dot product graph as an extension of the multi-RDPG. This model provides the framework for a multi-RDPG approach to a collection of weighted graphs.

2. We show that the model can be fitted in the case of both Poisson distributed weights and normally distributed weights with equal variances.

3. We develop an a test for whether multiple graphs are drawn from the same distribution in the cases of Poisson and normally distributed weights.

The paper is organized as follows: In Section 2, we present the weighted multiple random dot product graph model; In Section 3, we consider the special case of Poisson distributed weights, develop a test for the null hypothesis that a number of graphs are drawn from the same distribution, and apply the test to a Wikipedia data set; In Section 4, we similarly consider the special case of normally distributed weights and apply the test to a sewage data set; Finally, we have the discussion in Section 5.

2 Generalized Multiple Random Dot Product Graph

We propose the following model which extends the multiple random dot product graph to K weighted graphs. Let ν be a probability distribution with L parameters,

$$A_{ij}^k \sim \nu(f_1((\Theta_1^k)_{ij}), \ldots, f_L((\Theta_L^k)_{ij})), \quad \Theta_l^k = U_l \Lambda_l^k U_l^T, \quad k = 1, \ldots, K, \quad l = 1, \ldots, L, \quad (4)$$

where $A_{ji} = A_{ij}\ \forall (i,j) \in \{1, \ldots, n\}^2$, U_l for $l = 1, \ldots, L$ are $n \times d$ orthogonal matrices, Λ_l^k for $l = 1, \ldots, L$ and $k = 1, \ldots, K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements, $f_l : \mathbb{R} \to S_l$ is a link function and S_l are the domains of the parameters $(\Theta_l^k)_{ij}$ for $l = 1, \ldots, L$ and $k = 1, \ldots, K$. Choosing ν to be the Bernoulli distribution the model reduces to the multi-RDPG (Nielsen and Witten, 2018).

3 Poisson Distribution

We now consider ν to be the Poisson distribution and recall that the Poisson distribution has one positive parameter which is also the mean (hence $L = 1$). The model (4) simplifies to,

$$A_{ij}^k \sim \text{Poisson}(f(\Theta_{ij}^k)), \quad \Theta^k = U \Lambda^k U^T, \quad k = 1, \ldots, K \quad (5)$$

where $A_{ji} = A_{ij}\ \forall (i,j) \in \{1, \ldots, n\}^2$, U is a $n \times d$ orthogonal matrix , Λ^k for $k = 1, \ldots, K$ are $d \times d$ diagonal matrices with nonnegative diagonal elements and $f : \mathbb{R} \to \mathbb{R}_+$. We will

only consider $f(x) = \max(0, x)$ because it is closely related to the identity.

3.1 Optimization

The optimization problem for fitting the model (5) is the same as for the multi-RDPG with f as the identity, because the Poisson distribution is also a one parameter distribution where the parameter is the mean which is exactly what is exploited by Nielsen and Witten (2018). Note that we ignore that the chosen f maps to the positive numbers by using the identity in the optimization problem but these functions are closely related. All elements of A^k are nonnegative so we expect it to rarely be a problem. Minimizing the sum of squares was motivated by the typical problem for fitting the random dot product graph which was in turn proposed as a computationally viable alternative to the maximum likelihood (Scheinerman and Tucker, 2010).

$$\underset{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda^1, \ldots, \Lambda^K \in \Delta_+^d}{\text{minimize}} \sum_{k=1}^{K} \|A_+^k - U \Lambda^k U^T\|_F^2, \tag{6}$$

where Δ_+^d is the set of $d \times d$ diagonal matrices with non-negative diagonal elements, $||\cdot||_F$ is the Frobenius norm, and A_+^k is the *positive semi-definite part* of A^k. That is if $A^k = VDV^T$ is the eigendecomposition of A^k where $D = \text{diag}(\alpha_1, \ldots, \alpha_n)$ is an $n \times n$ diagonal matrix then $A_+^k = VD_+V^T$ where $D_+ = \text{diag}(\max(0, \alpha_1), \ldots, \max(0, \alpha_n))$ is an $n \times n$ diagonal matrix is the positive semi-definite part of A^k. The adjacency matrices of the weighted and undirected graphs are symmetric as is required by the algorithm by Nielsen and Witten (2018) for fitting multi-RDPG and we will therefore fit (5) using their algorithm.

3.2 A Test for $H_0 : \Lambda^1 = \cdots = \Lambda^K$

We consider a test for the null hypothesis that all of K observed graphs are drawn from the same distribution. In the model (5) that corresponds to the null hypothesis,

$$H_0 : \Lambda^1 = \cdots = \Lambda^K \tag{7}$$

Following the approach laid out in Nielsen and Witten (2018) we consider a likelihood ratio inspired test statistic which is the difference in best fits in terms of sum of squares for a model where $\Lambda^1 = \cdots = \Lambda^K$ are assumed equal and in a model where they are not. The test statistic, which turns out exactly parallel to Nielsen and Witten (2018), is,

$$T\left(A_+^1, \ldots, A_+^K\right) = \left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda \in \Delta_+^d} \sum_{k=1}^{K} ||A_+^k - U\Lambda U^T||_F^2\right) - \left(\min_{U \in \mathbb{R}^{n \times d},\, U^T U = I,\, \Lambda^1, \ldots, \Lambda^K \in \Delta_+^d} \sum_{k=1}^{K} ||A_+^k - U\Lambda^k U^T||_F^2\right). \tag{8}$$

We notice that this will be zero under the null hypothesis and that large values are critical. The second term of (8) is the minimum of (6) and can be computed using the algorithm by Nielsen and Witten (2018) as noted in Section 3.1. To compute the first term we consider the model where the diagonal values are the same for all graphs ($\Lambda^k = \Lambda \, \forall k$).

$$A_{ij}^k \sim \text{Poisson}(f(\Theta_{ij})), \quad \Theta = U\Lambda U^T, \quad k = 1, \ldots, K \tag{9}$$

where $A_{ji} = A_{ij} \; \forall (i,j) \in \{1, \ldots, n\}^2$, U is a $n \times d$ orthogonal matrix, Λ is a $d \times d$ diagonal matrix with nonnegative diagonal elements and $f : \mathbb{R} \to \mathbb{R}_+$. Because the optimization problem to fit (9) also becomes analogous to the problem under the null in Nielsen and Witten (2018) we can use that they showed that the parameters of this model are estimated by the eigen decomposition of $\frac{1}{K}\sum_{k=1}^{K} A_+^k = QDQ^T$. If the diagonal elements of D are in non-increasing order then let U be the first d columns of Q and let Λ contain the first d diagonal values of Q.

$T(A_+^1, \ldots, A_+^K)$ is compared to its null distribution to compute a p-value. The null distribution is obtained by parametrically sampling from the fitted null model (9). Details are in Algorithm 1.

3.2.1 Simulation Study

A simulation study is conducted to show that the p-values are uniformly distributed under the null hypothesis which indicates type I error control, and that the test has adequate power under the alternative.

Algorithm 1 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (5)

1. Fit the model in (9) by finding the eigen decomposition $\frac{1}{K}\sum_{k=1}^{K} A_+^k = QDQ^T$ where the diagonal elements of D are in non-increasing order. Then let $\hat{U}$ be the first d columns of Q and $\hat{\Lambda}$ contain the first d diagonal values of Q
2. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (8).
3. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{k,*b}\right)_{ij}$ be generated by simulating from the distribution Poisson$(f((\hat{U}\hat{\Lambda}\hat{U}^T)_{ij}))$ for all $k = 1, \ldots, K$
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (8).
4. Compute the p-value,

$$p = \frac{1}{B}\sum_{b=1}^{B} I_{T\left(A_+^{1,*b},\ldots,A_+^{K,*b}\right) \geq T\left(A_+^1,\ldots,A_+^K\right)},$$

where I_C is an indicator function that equals one if the event C is true, and equals zero otherwise.

Type I Error We simulate two examples of data under the null hypothesis. We generate $K = 2$ graphs with

$$U = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & -1 & 1 & -1 & \dots & -1 \end{bmatrix}^T \tag{10}$$

and $\Lambda^1 = \Lambda^2 = \text{diag}(10n, 5n)$ for $n \in \{20, 50, 100\}$. We generate the adjacency matrices A^k according to (5) with f as the identity for $k = 1, 2$. A p-value is then computed according to Algorithm 1 with $d = 2$. Data is simulated this way 1,000 times and we obtain 1,000 p-values (Figure 1a).

The procedure is repeated using $d = 3$, $\Lambda^1 = \Lambda^2 = \text{diag}(2n, n/2, n)$ and

$$U = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & -1 & 1 & -1 & \dots & -1 \\ 1 & 1 & -1 & -1 & \dots & -1 \end{bmatrix}^T . \tag{11}$$

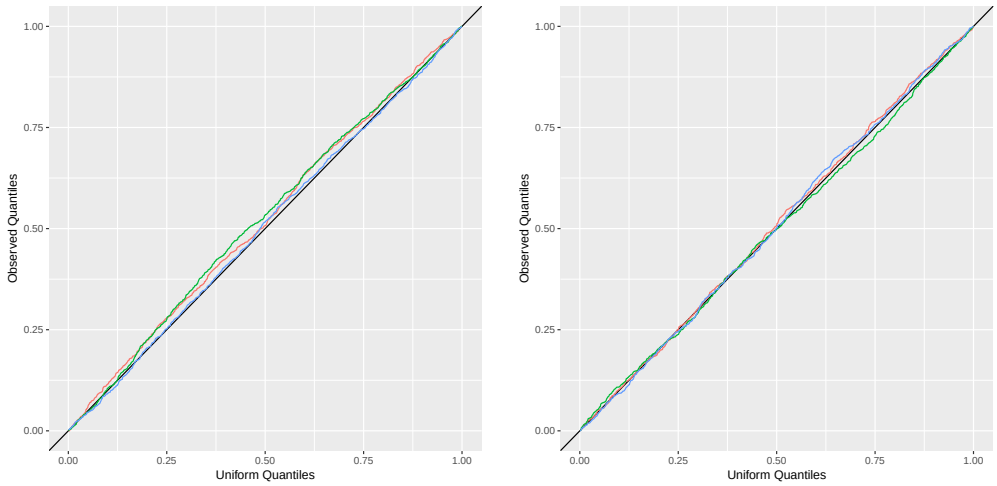
The p-values are shown in Figure 1b.

In Figure 1a the lines corresponding to $n \in \{20, 50\}$ seem to be slightly above the diagonal but when n increases to 100 the line looks close to the diagonal. In Figure 1b all lines look close to the diagonal. In general, the empirical distribution of the p-values are satisfactorily close uniformly distributed in both panels of Figure 1 indicating adequate type I error control.

Power We generate data for which the hypothesis $H_0 : \Lambda^1 = \cdots = \Lambda^k$ is not true, in order to explore the power of the test in Algorithm 1. For $r \in [0, 1]$ and $n \in \{10, 20, 50\}$ we generate Λ^1 and Λ^2 as follows,

$$\Lambda^1 = \text{diag}\left(2n(1+r), 2n(1-r)\right), \quad \Lambda^2 = \text{diag}\left(2n(1-r), 2n(1+r)\right). \tag{12}$$

U is generated as in (10). A^1 and A^2 are generated according to (5). A p-value is calculated using Algorithm 1. This is repeated 1,000 times for each value of r leading to p-values



(a) (b)

Figure 1: Quantile-quantile plots of p-values from Algorithm 1 against a uniform distribution. The colors indicate the value of n: $n = 20$ (red), $n = 50$ (green), and $n = 100$ (blue). *(a):* Data were generated according to (5) with $\Lambda^1 = \Lambda^2 = \text{diag}(10n, 5n)$ and U as in (10). *(b):* Data were generated according to (5) with $\Lambda^1 = \Lambda^2 = \text{diag}(2n, n/2, n)$ and U as in (11).

$p_1, \ldots, p_{1000}$. The power is then computed at level $\alpha = 0.05$ as

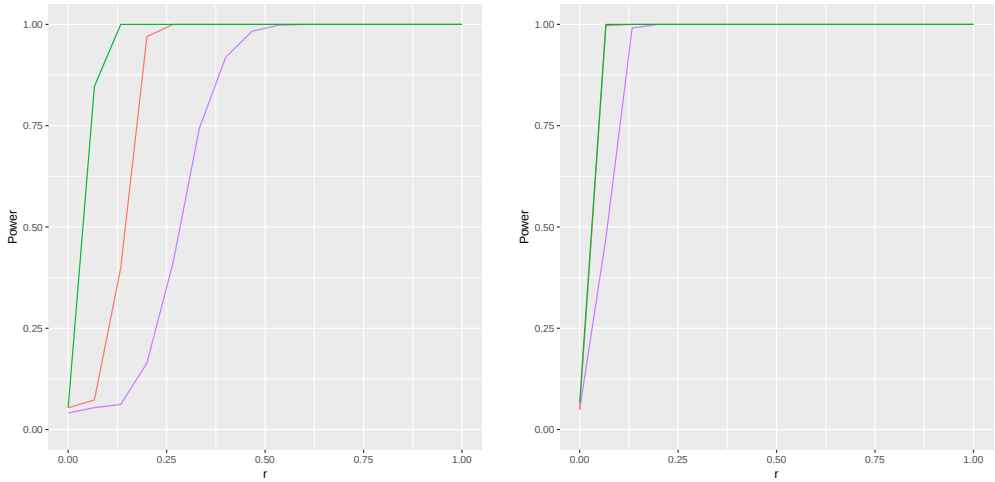
$$\text{power} = \frac{1}{1000} \sum_{i=1}^{1000} I_{p_i < \alpha}, \tag{13}$$

where I_C is an indicator function for the event C. The power is shown in Figure 2a and as expected the power increases with r and n.

We repeat this procedure with U as in (11) and with

$$\Lambda^1 = \text{diag}\left(10n(1-r), 5n(2+r), n(1-r)\right), \quad \Lambda^2 = \text{diag}\left(10n(1+r), 5n(2-r), n(1+r)\right). \tag{14}$$

We see again that the power increases with r and n (Figure 2b).



(a) Λ_1, Λ_2 in (12), and U in (10)

(b) Λ_1, Λ_2 in (14), and U in (11)

Figure 2: The power of the test in Algorithm 1. The colors indicate the value of n: $n = 10$ (purple), $n = 20$ (red), and $n = 50$ (green). *(a):* Λ_1, Λ_2 are defined in (12), and U in (10) *(b):* Λ_1, Λ_2 are defined in (14), and U in (11)

3.2.2 Application to Wikipedia Data

We consider a data set which represents a subset of Wikipedia articles as graphs (Suwan et al., 2016). The data set is accessible at `http://www.cis.jhu.edu/~parky/Data/data.html`. It consists of 1383 articles in both English and French represented as nodes in the graphs. The graph representing the English and respectively French Wikipedia subset is constructed by having an edge between two nodes if one (or both) of the corresponding articles links to the other and then weighing the edge with the number of links. The edges weights in the English graph are scaled such that the average of the edge weights is equal to the average of the edge weights in the French graph. Because there is a higher number of links in the English graph and we are interested in the structure of the links, we need to scale the larger graph.

We test the hypothesis $H_0 : \Lambda^{\text{English}} = \Lambda^{\text{French}}$ using Algorithm 1 with $d = 5$. This gives a test statistic value of $T(A_+^{\text{English}}, A_+^{\text{French}}) = 11018.8$ and a p-value of $p = 0$. We therefore reject the hypothesis that the two graphs are drawn from the same distribution.

4 Normal Distribution

Assuming normally distributed weights, we have two parameters to consider when fitting the model. We will assume equal variance for all weights across graphs and that these weights are independent. The model diverges from definition (4) as we will not parameterize the variance as a product. The model is thus,

$$A_{ij}^k \sim \text{Normal}(f(M_{ij}^k), \sigma^2), \quad M^k = U\Lambda^k U^T, \quad \sigma^2 \geq 0, \quad k = 1, \ldots, K \tag{15}$$

where $A_{ji} = A_{ij}$ $\forall (i,j) \in \{1, \ldots, n\}^2$ and $f : \mathbb{R} \to \mathbb{R}$ is a link function. We will only consider f to be the identity.

4.1 Fitting

Choosing the link function f to be the identity the optimization problem to estimate the parameters for the mean again reduces to minimizing the sum of squares in (6) and we can use the algorithm by Nielsen and Witten (2018) to fit it. For simplicity will we only consider the estimation of the mean parameter in the model (15). We will, however, need to estimate it under the null hypothesis.

4.2 A Test for $H_0 : \Lambda^1 = \cdots = \Lambda^K$

Having assumed equal variances we aim to test the null hypothesis thatl K graphs are drawn from the same distribution. That is,

$$H_0 : \Lambda^1 = \cdots = \Lambda^K. \tag{16}$$

The test statistic for H_0 is, as was the case for Poisson distributed weights, given by (8). The test algorithm has the same structure as Algorithm 1, but as we simulate from a normal distribution we need to also estimate the variance under the null hypothesis in order to perform the parametric sampling.

Under the null hypothesis the model is,

$$A_{ij}^k \sim \text{Normal}(f(M_{ij}), \sigma^2), \quad M = U\Lambda U^T, \quad k = 1, \ldots, K. \tag{17}$$

We have assumed that all elements are normally distributed and that they all have the same variance. Consider $Z \sim N_q(\mu, \sigma^2 I_q)$ where $\mu \in \mathbb{R}^q$, $\sigma^2 > 0$, and $q \in \mathbb{N}_+$. The unbiased estimate of the variance is known to be

$$\tilde{\sigma}^2 = \frac{1}{q-p}\sum_{i=1}^{N}(z_i - \hat{\mu}_i)^2 \tag{18}$$

where p is the number of parameters estimated in the means, $\hat{\mu}$ is the estimated mean, and z_i are the observations for $i = 1, \ldots, q$. We have $q = \frac{K}{2}(n^2 + n)$ observations as the adjacency matrices are symmetric. The means are estimated as $\hat{M} = \hat{U}\hat{\Lambda}\hat{U}^T$which has $nd - \frac{d(d+1)}{2}$ parameters estimated in the orthogonal matrix $\hat{U}$ and d in the diagonal matrix $\hat{\Lambda}$. This gives a total of $p = nd - \frac{d(d-1)}{2}$ parameters estimated in the means and the unbiased estimate of the variance is,

$$\tilde{\sigma}^2 = \left(\frac{K}{2}(n^2 + n) - nd + \frac{d(d-1)}{2}\right)^{-1} \sum_{k=1}^{K}\sum_{i=1}^{n}\sum_{j=1}^{i}(A_{ij}^k - \hat{M}_{ij})^2. \tag{19}$$

The test algorithm is found in Algorithm 2.

4.2.1 Simulation Study

We again conduct a simulation study to show that the p-values are uniformly distributed which indicates type I error control, and that the test has adequate power under the alternative.

Type I Error The procedure from Section 3.2.1 is repeated using U as in (10), $\Lambda^1 = \Lambda^2 = \text{diag}(10n, 5n)$, and $\sigma^2 = 1$ for $n \in \{50, 75, 100\}$. The adjacency matrices A^k are generated according to (15) and the p-value is computed using Algorithm 2. The p-values are shown in Figure 3a. This procedure is repeated using U as in (11), $\Lambda^1 = \Lambda^2 = \text{diag}(50n, 25n, n/4)$, and $\sigma^2 = 1$ (Figure 3b).

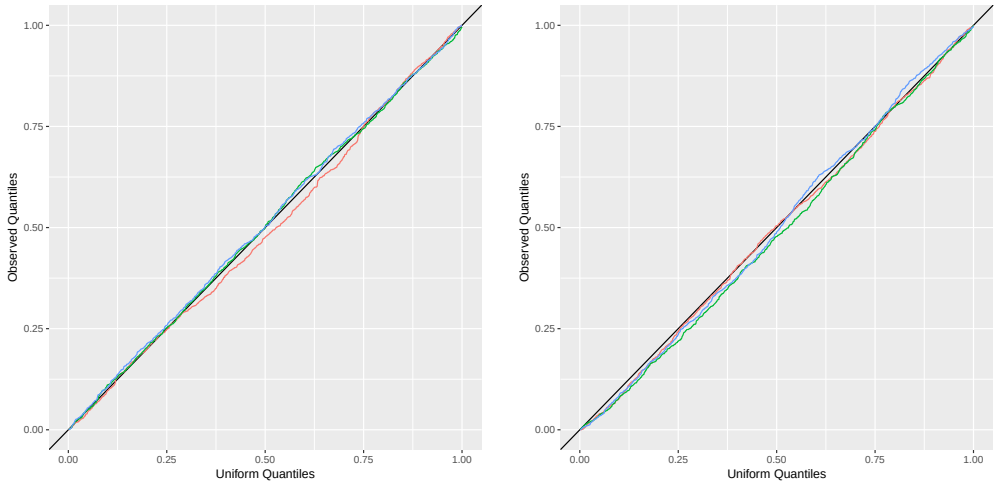
Algorithm 2 A Test for $H_0 : \Lambda^1 = \ldots = \Lambda^K$ in (15)

1. Estimate the mean parameter in (17) by finding the eigendecomposition $\frac{1}{K}\sum_{k=1}^{K} A_+^k = QDQ^T$ where the diagonal elements of D are in non-increasing order. Then let $\hat{U}$ be the first d columns of Q and $\hat{\Lambda}$ contain the first d diagonal values of Q
2. Estimate the variance parameter in (17) according to (19)
3. Compute $T\left(A_+^1, \ldots, A_+^K\right)$ according to (8).
4. For $b = 1, \ldots, B$:
   (a) Generate $A_+^{1,*b}, \ldots, A_+^{K,*b}$ as follows:
      i. For all $i \leq j$, let $\left(A^{k,*b}\right)_{ij}$ be generated by simulating from the distribution Normal$((\hat{U}\hat{\Lambda}\hat{U}^T)_{ij}, \tilde{\sigma}^2)$ for all $k = 1, \ldots, K$
      ii. For all $i < j$ and all $k = 1, \ldots, K$, set $\left(A^{k,*b}\right)_{ji}$ equal to $\left(A^{k,*b}\right)_{ij}$.
      iii. Let $A_+^{1,*b}, \ldots, A_+^{K,*b}$ be the positive semi-definite parts of $A^{1,*b}, \ldots, A^{K,*b}$.
   (b) Compute $T\left(A_+^{1,*b}, \ldots, A_+^{K,*b}\right)$ according to (8).
5. Compute the p-value,

$$p = \frac{1}{B}\sum_{b=1}^{B} I_{T\left(A_+^{1,*b},\ldots,A_+^{K,*b}\right) \geq T\left(A_+^1,\ldots,A_+^K\right)},$$

where I_C is an indicator function that equals one if the event C holds, and equals zero otherwise.

In both panels of Figure 3, the empirical distributions of the p-values are satisfactorily close to uniformly distributed indicating adequate type I error control.

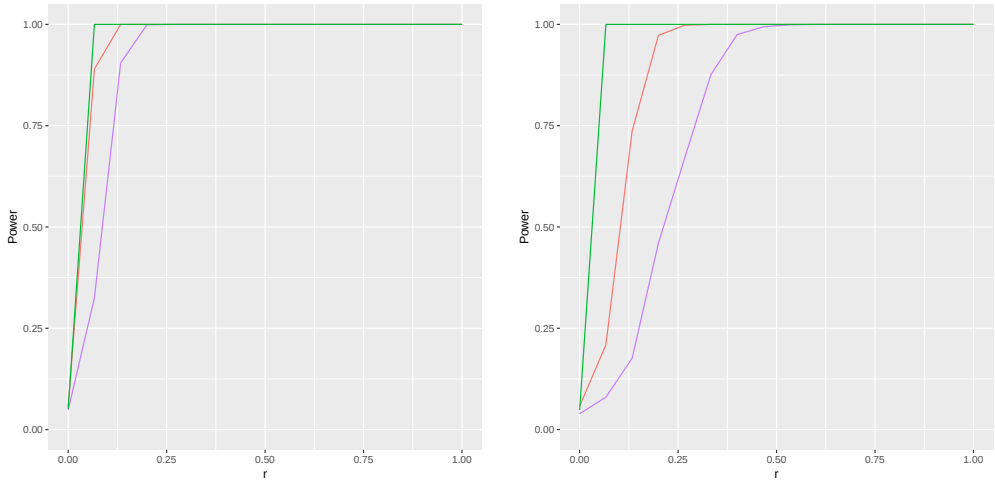


(a) (b)

Figure 3: Quantile-quantile plots of p-values from Algorithm 1 against a uniform distribution. The colors indicate the value of n: $n = 50$ (red), $n = 75$ (green), and $n = 100$ (blue). *(a):* Data were generated according to (15) with $\Lambda^1 = \Lambda^2 = \text{diag}(10n, 5n)$, U as in (10), and $\sigma^2 = 1$. *(b):* Data were generated according to (15) with $\Lambda^1 = \Lambda^2 = \text{diag}(50n, 25n, n/4)$, U as in (11), and $\sigma^2 = 1$.

Power As in Section 3.2.1, we here generate data for which the hypothesis does not hold to evaluate the capacity of the test to reject false hypotheses. For $r \in [0, 1]$ and $n \in \{10, 20, 50\}$ we generate Λ^1 and Λ^2 as in (12), U as in (11), and $\sigma^2 = 1$. A^1 and A^2 are generated according to (15) and a p-value is computed according to Algorithm 2. This is repeated 1,000 times and the power is calculated as in (13). As seen in Figure 4a the power increases with r and n.

This procedure is repeated with U as in (11) and Λ^1 and Λ^2 as in (14). In Figure 4b we again see that the power increases with r and n.



(a) Λ_1, Λ_2 in (12), and U in (10)

(b) Λ_1, Λ_2 in (14), and U in (11)

Figure 4: The power of the test in Algorithm 2. The colors indicate the value of n: $n = 10$ (purple), $n = 20$ (red), and $n = 50$ (green). *(a):* Λ_1, Λ_2 are defined in (12), and U in (10) *(b):* Λ_1, Λ_2 are defined in (14), and U in (11)

4.2.2 Application to Oribatid Mites

We consider a data set consisting of counts of Oribatid mites in 70 different soil cores in different locations (Borcard et al., 1992; Borcard and Legendre, 1994, 2002; Oksanen et al., 2017). It consists of the number of mites of 35 different species present in the cores. We construct networks by letting the sampling sites (soil cores) be the nodes and letting the edge weights be the difference in population composition at the corresponding sampling sites in terms of Bray-Curtis dissimilarity (Bray and Curtis, 1957),

$$BC_{ij} = \frac{\sum_s |y_{si} - y_{sj}|}{\sum_s (y_{si} + y_{sj})}, \tag{20}$$

where y_{si} is the number of species s present at sample site i. In order to construct two networks we split the species into two groups: The first 17 species in the data set in one group and the remaining 18 species in the other. The dissimilarities are calculated within each group. We test the hypothesis $H_0 : \Lambda^1 = \Lambda^2$ using Algorithm 2. We get a value of the test statistic to be $T(A^1_+, A^2_+) = 1.55$ and a p-value of $p = 0.037$. We therefore reject the

hypothesis that the two networks are drawn from the same distribution. We refrain from making biological interpretations.

5 Discussion

In this paper we propose a generalization of the multi-RDPG to weighted graphs. Being able to handle weighted graphs opens the framework up to more practical applications. We propose approaches for testing whether a number of graphs on the same set of nodes are drawn from the same distribution in the cases of both Poisson distributed weights and normally distributed weights with the variance assumed equal for all weights across graphs. Both approaches control type I error and has adequate power against the alternative.

We have considered the link function to be the identity which is natural in the case of normally distributed weights. In the case of Poisson distributed weights, a link function assuring that the parameter is positive would be more natural.

Acknowledgments

We thank our colleagues Bjarne Ersbøll, Nina Munkholt Jakobsen, and Sofie Pødenphant Jensen for fruitful discussions.

References

Aicher, C., A. Z. Jacobs, and A. Clauset (2013). Adapting the stochastic block model to edge-weighted networks. *arXiv preprint arXiv:1305.5782*.

Borcard, D. and P. Legendre (1994). Enviromental control and spatial structure in ecological comminities: an example using oribatid mites (acari,oribatid). *Enviromental and Ecological Statistics 1*, 37–61.

Borcard, D. and P. Legendre (2002). All-scale spetial analysis of ecological data by means of principal coordinates of neighbour matrices. *Ecological Modelling 153*, 51–68.

Borcard, D., P. Legendre, and P. Drapeau (1992). Partialling out the spatial component of ecological variation. *Ecology 73*, 1045–1055.

Bray, J. R. and J. Curtis (1957). An ordination of upland forest communities of southern wisconsin. *Ecological Monographs 27*, 325–349.

DeFord, D. R. and D. N. Rockmore (2016). A random dot product model for weighted networks. *arXiv preprint arXiv:1611.02530*.

Dong, X., P. Frossard, P. Vandergheynst, and N. Nefedov (2014). Clustering on multi-layer graphs via subspace analysis on grassmann manifolds. *IEEE Transactions on Signal Processing 62*(4), 905–918.

Durante, D., D. B. Dunson, and J. T. Vogelstein (2017). Nonparametric Bayes modeling of populations of networks. *Journal of the American Statistical Association 112*(520), 1516–1530.

Erdös, P. and A. Rényi (1959). On random graphs, i. *Publicationes Mathematicae (Debrecen) 6*, 290–297.

Garlaschelli, D. (2009). The weighted random graph model. *New Journal of Physics 11*(7), 073005.

Hoff, P. (2008). Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, pp. 657–664.

Hoff, P. D. (2009). Multiplicative latent factor models for description and prediction of social networks. *Computational and Mathematical Organization Theory 15*(4), 261.

Hoff, P. D., A. E. Raftery, and M. S. Handcock (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association 97*(460), 1090–1098.

Holland, P. W., K. B. Laskey, and S. Leinhardt (1983). Stochastic blockmodels: First steps. *Social Networks 5*(2), 109–137.

Ma, Z. and Z. Ma (2017). Exploration of large networks via fast and universal latent space model fitting. *arXiv preprint arXiv:1705.02372*.

Nickel, C. L. M. (2008). *Random Dot Product Graphs A Model For Social Networks*. Ph. D. thesis, The Johns Hopkins University.

Nielsen, A. M. and D. Witten (2018). The multiple random dot product graph. *Submitted to Journal of Graphical and Computational Statistics*.

Oksanen, J., F. G. Blanchet, M. Friendly, R. Kindt, P. Legendre, D. McGlinn, P. R. Minchin, R. B. O'Hara, G. L. Simpson, P. Solymos, M. H. H. Stevens, E. Szoecs, and H. Wagner (2017). *vegan: Community Ecology Package*. R package version 2.4-5.

Scheinerman, E. R. and K. Tucker (2010). Modeling graphs using dot product representations. *Computational Statistics 25*(1), 1–16.

Shiga, M. and H. Mamitsuka (2012). A variational Bayesian framework for clustering with multiple graphs. *IEEE Transactions on Knowledge and Data Engineering 24*(4), 577–590.

Suwan, S., D. S. Lee, R. Tang, D. L. Sussman, M. Tang, C. E. Priebe, et al. (2016). Empirical Bayes estimation for the stochastic blockmodel. *Electronic Journal of Statistics 10*(1), 761–782.

Tang, R. (2017). *Robust Estimation from Multiple Graphs*. Ph. D. thesis, Johns Hopkins University.

Tang, W., Z. Lu, and I. S. Dhillon (2009). Clustering with multiple graphs. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pp. 1016–1021. IEEE.

Wang, S., J. T. Vogelstein, and C. E. Priebe (2017). Joint embedding of graphs. *arXiv preprint arXiv:1703.03862*.

Wu, Y.-J., E. Levina, and J. Zhu (2017). Generalized linear models with low rank effects for network data. *arXiv preprint arXiv:1705.06772*.

Young, S. J. and E. R. Scheinerman (2007). Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pp. 138–149. Springer.

Appendix C

Comparison of Methods for Disease Progression Prediction

Nielsen, A. M., Nielsen, R. L., Donnelly, L., Zhou, K., Dahl, A. B., Gupta, R., Ersbøll, B. K., Pearson, E., Clemmensen, L. (2018) *A Comparison of Methods for Disease Progression Prediction Through a GoDARTS Study*, (draft intended for a medicine methodology journal)

A Comparison of Methods for Disease Progression Prediction Through a GoDARTS Study

Agnes M Nielsen*, Rikke L Nielsen*, Louise Donnelly†,
Kaixin Zhou†, Anders B Dahl*, Ramneek Gupta*,
Bjarne K Ersbøll*, Ewan Pearson†, Line Clemmensen*

*Technical University of Denmark
†University of Dundee

December 26, 2018

Abstract

In recent years, a variety of new machine learning methods are being employed in prediction of disease progression, e.g. random forest or neural networks, but how do they compare to and are they direct substitutes for the more traditional statistical methods like the Cox proportional hazards model? In this paper, we compare three of the most commonly used approaches to model prediction of disease progression.

We consider a case from a cohort-based population in Tayside, UK. In this study, the time until a patient goes onto insulin treatment is of interest; in particular discriminating between slow and fast progression. This means that we are both interested in the results as a raw time-to-insulin prediction but also in a dichotomized outcome making the prediction a classification.

Three different methods for prediction are considered: A Cox proportional hazards model, random forest for survival data and a neural network on the dichotomized outcome. The performance is evaluated using survival performance measures (concordance indices and the integrated Brier score) and using the accuracy, sensitivity, specificity, and Matthews correlation coefficient for the corresponding classification problems.

We discuss the limitations of the three approaches and where they each excel in terms of prediction performance, interpretation, and how they handle data imbalance.

Keywords:

1 Background

In the medical literature there are typically three approaches to modeling a time-to-event study where it is of interest to predict the time for which a patient e.g. goes on insulin. The first approach is to use parametric or semi-parametric models of which the most common is the Cox proportional hazards model, which directly models the time to event for each subject (Cox, 1972). Limiting ourselves to diabetes literature, as our case concerns diabetes, this approach has for example been used to predict incident diabetes in Sattar et al. (2004), the risk of diabetes in Hippisley-Cox et al. (2009), incidents of cardiovascular events, diabetes and death in Liljestrand et al. (2015), risk of type 2 diabetes in Hippisley-Cox and Coupland (2017), and progression to diabetes in Steck et al. (2018).

The second approach is to use newer developments like decision trees or random forest survival models (Hothorn et al., 2004, 2006; Ishwaran et al., 2008). These model the survival time in a non-parametric way by splitting the data into smaller groups. A survival tree has been used to analyse disease progression in type 1 diabetes in Xu et al. (2016). Random forests for survival data have for example been used to predict the risk of diabetes complications (Lagani et al., 2015) and the risk of diabetic retinopathy (Semeraro et al., 2011).

The third approach is to predict a given time step ahead, e.g. 1, 2, or 5 years, by use of simpler linear models or machine learning methods. Linear regression models have for example been used to study 1-year HbA1c reduction (Farmer et al., 2015) and predict asymmetric dimethylarginine levels in patients with type 2 diabetes mellitus (Ganz et al., 2017). Additionally, logistic regression has been used to predict (classify) diabetes and cardiovascular disease at the time of the study (Janiszewski et al., 2007), drug-treated diabetes diagnosed during 5-year follow-up (Lindström and Tuomilehto, 2003), 5-year diabetes risk after gestational diabetes mellitus in Claesson et al. (2017), and prediction of highest quartiles of asymmetric dimethylarginine levels in patients with type 2 diabetes mellitus (Ganz et al., 2017). Linear regression and logistic regression models come with the additional advantage of hypothesis tests for the covariate coefficients. Machine learning methods on the other hand often give rise to better predictions due to handling non-linearities as well as co-linearities typically at the cost of missing out on hypothesis testing for individual

covariates. This makes the models harder to interpret. Machine learning methods have seen an increased use in the recent years as for example in Dagliati et al. (2018) where several methods including support vector machines and random forests have been used to predict diabetes complications at 3, 5, and 7 years from the first visit. They find that random forest performs the best but choose logistic regression for application in the clinic due to it being easier to interpret. Recently, there have been efforts in interpretation of machine learning methods (Welling et al., 2016; Ribeiro et al., 2016) and tests for inclusion of variables in a random forest (Mentch and Hooker, 2017; Wager and Athey, 2018).

We compare these three approaches to elucidate their pros and cons for the analysis of medical data, in particular with focus on diabetes, but we believe this comparison is a useful basis for other medical areas with similar cohort studies. We do this by reviewing one widely applied method within each of these approaches. We then apply the methods to a diabetes data set. The data set investigates the time until a type 2 diabetes patient goes onto insulin from the day of diagnosis. It consists of around 7000 patients and has both clinical and biochemical variables. The first method is Cox proportional hazards model which has previously been applied to this data set (Doney et al., 2004, 2005; Zhou et al., 2014); the second is a random forest approach for survival data; and the last is a neural network which has also been applied in this data set in Nielsen et al. (2018). We consider two versions of the data set: Using the biochemical values closest to the diagnosis and using features extracted from one year around diagnosis. This is done to investigate if this is a good feature extraction strategy and to have two versions of the data set for the method comparisons. Finally, we give recommendations based on this study.

The rest of the paper is organised as follows. In Section 2, we present the data set as well as the methods used in the analysis. The results of the analysis of the data set are presented in Section 3, a discussion of the methods is given in Section 4, and conclusions based on the study are given in Section 5.

Table 1: Summary of biochemical variables with longitudinal measurements. The table shows the median and quantiles (first and third) or the number of measurements per person (MPP), the mean and standard deviation of the baseline measurement (Baseline), number of people with a baseline measurement (Total), and number of people with more than three measurements within the year around diagnosis (total w. $\geq$ 3 m.).

Variable	MPP	Baseline	Total	Total w. $\geq$ 3 m.
Alanine transaminase	1 (0-2)	36.39 (33.60)	4261	1376
Aspartate aminotransferase	0 (0-0)	28.74 (26.94)	156	31
Cholesterol	2 (1-3)	5.36 (1.34)	5546	1785
Creatinine	2 (1-4)	79.63 (23.42)	5702	2994
Glycated haemoglobin (HbA1c)	2 (1-3)	8.37 (2.03)	5842	2113
High-density lipoprotein	2 (1-3)	1.19 (0.33)	5317	1645
Low-density lipoprotein	0 (0-1)	2.86 (1.03)	3136	159
Triglycerides	1 (0-1)	3.08 (3.14)	3693	297

2 Methods

2.1 Data

We study the medical records of 6871 patients with type 2 diabetes from the Genetics of Diabetes Audit and Research (GoDARTS) database (Doney et al., 2004, 2005; Zhou et al., 2014).

The data consist of biochemical markers which were obtained during regular patient visits and have been recorded at different times and with varying frequency for each patient (Table 1), clinical variables including anthropometric, life-style and drug prescription variables (Table 2), and the time from diagnosis to first insulin treatment or they left the study as well as whether insulin treatment was given. In this data set, there is around 58% censoring meaning patients who did not receive insulin treatment while participating in the study. We only consider patients with type 2 diabetes. In order to minimize the number of type 1 diabetes patients, only patients diagnosed after 35 years of age are included. Besides this only patients diagnosed in 2010 or earlier are included. This leaves 6324 patients.

Table 2: Summary of clinical variables. The table shows the mean and standard deviation of the variables or for categorical variables the number of observations in each category, and the total number of observations.

Variable	Mean (SD) or totals	Total
BMI (kg/m^2)	31.56 (6.23)	5139
Gender		6324
Female	2803	
Male	3521	
Smoking		6324
No	1593	
Yes	4731	
Social class	3 (2-4)*	6229
Age at diagnosis (days)	22574 (3947.75)	6324
Year of diagnosis	2002 (1999-2005)*	6324
Treatment at diagnosis		6324
None or missing	4924	
Mono	1354	
Dual	46	
Weight (kg)	88.22 (19.25)	5139
Diastolic blood pressure	82.19 (11.02)	5720
Systolic blood pressure	143.3 (20.15)	5720
Glutamic acid decaboxylase (U/ml)		4860
<11	4655	
>11	205	

*Shows the median and the quantiles.

2.2 Feature Extraction

We define two data sets extracted from the biochemical markers listed in Table 1. The first is the *baseline data set* which consists of one measurement per variable j and person i. It is the measurement closest to the day of confirmed diagnosis (HbA1c>6.5%) and within plus or minus 182.5 days (Table 1).

The second data set, the *time model data set*, is created as follows. For each variable j we consider all measurements within 182.5 days of diagnosis and extract three features describing the time-dependent behavior. A linear trend is estimated by solving (Eckner, 2012).

$$\underset{a_0^j, a_1^j}{\text{minimize}} \sum_{i=2}^{n} (t_i^j - t_{i-1}^j)(x_i^j - a_0^j - a_1^j t_i^j)^2 \tag{1}$$

where t_i^j is the i'th time point for variable j, x_i^j is the i'th measurement value of variable j, and a_0^j and a_1^j are the two parameters variable j. Let $z_i^j = x_i^j - a_0^j - a_1^j t_i^j$ and we fit a first order auto regressive model by solving (Mudelsee, 2014).

$$\underset{\tau^j}{\text{minimize}} \sum_{i=2}^{n} \left(z_i^j - \exp\left(\frac{-(t_i^j - t_{i-1}^j)}{\tau^j} \right) z_{i-1}^j \right)^2 \tag{2}$$

where τ^j is the auto-regressive parameter. This gives three extracted features for each biochemical variable j: $\hat{a}_0^j$ describing the general level, $\hat{a}_1^j$ describing a linear trend, and $\hat{\tau}^j$ capturing the auto regressive aspect. These variables are extracted for all biochemical variables that have at least 3 measurements.

2.3 Prediction

A Cox proportional hazard model (Cox, 1972) and a random forest model for survival (also known as the conditional inference forest) (Breiman, 2001; Hothorn et al., 2006; Strobl et al., 2007, 2008) are fitted to the survival data $\{X_i, T_i, \delta_i\}_{i=1}^n$ where X_i contains the explanatory variables, T_i is the time to event and δ_i is the event status, i.e. whether the event happens within the study period ($\delta_i = 1$) or not ($\delta_i = 0$).

Besides this, a neural network is fitted to the dichotomized data $\{X_i, Y_i\}_{i=1}^n$ where $Y_i = I(T_i < t_c \wedge \delta_i = 1)$, t_c is the cut off, and I is the indicator function, i.e. Y_i indicates

whether the event happens before a set time point t_c.

2.3.1 Cox Proportional Hazards Model

The Cox proportional hazards model is given by Cox (1972) as

$$\lambda(t|X_i) = \lambda_0(t)\exp(X_i\beta) \tag{3}$$

which models the hazard at time t (event rate at time t given survival until at least time t) for subject i with the explanatory variables X_i. Here β are the model parameters and λ_0 is the unknown base hazard. From this model it is not possible to give the predicted survival times because the base hazard is unknown. The predictions can therefore be given as either the linear predictors in the test data set ($X_{\text{new}}\hat{\beta}$) where $\hat{\beta}$ is the estimate of the β; or as an estimate of the survival function $\hat{S}(t)$. The model is fitted using the *rms* package in R (Harrell Jr, 2018; R Core Team, 2017).

2.3.2 Random Forest for Survival Data

The random forest model fits a number of trees to form a forest (Breiman, 2001). A tree is created by recursively splitting the data into smaller subsets based on some criteria (see example in Figure 1). The starting point which contains all the data is called the root node. When the data is split it forms two child nodes. These can also be split into child nodes meaning that the subset of data is split again. This can continue until only one observation is in each child node or until a stopping criteria is reached. The final nodes are called terminal nodes. Trees are unbiased predictors but have high variance. Other advantages are that they can handle mixed variables and missing data, and that they can do variable selection.

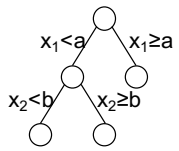


Figure 1: Example of a tree where the data is first split on variable x_1 and then on x_2.

A forest is created by bootstrapping the data, and building a single tree for each bootstrap sample of the data set in a process called bagging (Algorithm 1) (Breiman, 1996). This is done to decorrelate the trees which lowers the variance of the predictions. The forest then inherits many of the advantages of trees but reduces the variance. In random forest by Breiman (2001) the trees are further decorrelated by selecting a variable at each split in a tree from a random subset of the variables.

Algorithm 1 Pseudo-algorithm for bagging of trees.

1. From 1 to number of trees
   (a) Take a bootstrap sample
   (b) Build tree on sample*
2. Aggregate results from all trees

The conditional inference forest (CIF) by Hothorn et al. (2004) which builds each tree testing a global hypothesis is chosen over the random survival forest by Ishwaran et al. (2008) which adheres strictly to the random forest conditions laid out by Breiman (2001), because it has been shown to perform as well or better (Nasejje et al., 2017).

A conditional inference tree is built by recursively repeating two steps (Hothorn et al., 2006) (Algorithm 2).

Algorithm 2 Pseudo-algorithm for conditional inference tree.

Let each node be defined by a non-negative case weight vector $\mathbf{w} \in \mathbb{R}_+^n$ such that observations which are elements of the node have non-negative weights and the weights are otherwise zero.
Repeat the following steps:

1. For case weights $\mathbf{w}$ test the global null hypothesis of independence between any of the variables and the response. If the hypothesis cannot be rejected stop and otherwise select the variable with the strongest association to the response.
2. Split the sample space of the chosen variable into two disjoint sets with each assigned to one node. The case weight vectors of the two new child nodes are calculated by multiplying each case weight with 1 if the observation is in the node's corresponding set and 0 otherwise.

The conditional inference forest consists of a number of conditional inference trees. The

prediction can either be given as an estimate of the survival time $\hat{T}_i$ or as an estimate of the survival function, $\hat{S}(t)$. The survival function is estimated by a Kaplan-Meier estimate on the aggregation of terminal nodes which a new observation falls in (Hothorn et al., 2004). The estimate of the survival time is calculated by the observation weighted average of the trees. The model can handle missing data by using surrogate splits meaning if a variable is missing it splits on another variable which leads to the same subsets (Hothorn et al., 2010). It is fitted using the R package *party* (Hothorn et al., 2018).

2.3.3 Neural Network

A single-hidden-layer neural network (NN) is used for the dichotomized response (Venables and Ripley, 2002). This model consists of three layers of artificial neurons (also known as units or nodes), input, hidden, and output layers (Figure 2). Each layer consists of a number of neurons that receives input from all neurons in the previous layer and sends the output to all neurons in the next layer. For binary classification the output layer only has one neuron. Neurons in the hidden and output layers process their inputs by multiplying by a weight, summing them, adding a constant and then taking a fixed activation function. The result is,

$$\hat{y} = \phi_0 \left(\alpha + \sum_h w_h \phi_h \left(\alpha_h + \sum_i w_{ih} X_i \right) \right) \tag{4}$$

where ϕ_0 is the output layer activation function, ϕ_h is the hidden layer activation function, w_h and w_{ih} are weights, α and α_h are constants, h runs over the hidden units, and i runs over the observations. This function is fitted for example using the logistic cost function as a loss function (in the two class case)

$$-\sum_{i=1}^{n} (Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i) + \lambda \sum_{ih} w_{ih}^2 \tag{5}$$

where $p_i = \exp(\hat{y}_i)/(\exp(\hat{y}_i) + \exp(1 - \hat{y}_i))$ and λ is a decay parameter. Y_i was defined previously as whether the event happened before a set time point.

The number of neurons in the hidden layer and the decay parameter are optimised in an inner cross-validation loop according to the area under the curve. Further, for this model all continuous variables are standardised by mean and standard deviation within the

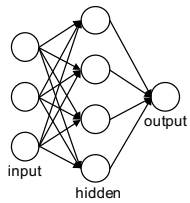


Figure 2: Example of a neural network.

cross-validation. This is fitted using the R package *nnet* (Ripley and Venables, 2016).

2.4 Missing Data Strategies

The Cox proportional hazards model cannot handle missing data. In the baseline data set, only the biochemical variables cholesterol, creatinine, HbA1c, and high-density lipoprotein are included with the clinical variables as the others have more than 50% missing values and all biochemical variables except high-density lipoprotein are log-transformed. Similarly, in the time model data, we only include extracted features with less than 50% missing values. These are the extracted features for cholesterol, creatinine, HbA1c, and high-density lipoprotein.

We have then removed all observations with missing values leaving only 1063 observations. This is done for simplicity. It is generally not advisable as it drastically reduces the number of observations and in cases where the missing data are not missing completely at random, the results are generally biased (Rubin, 1976; Donders et al., 2006). However, the alternatives of imputing the missing data using single imputation often causes us to be overly certain about the results and more advanced multiple imputation methods are generally better but also have their own problems (Sterne et al., 2009).

The neural network model used here cannot handle missing data either. In this case, we use the imputation strategy of replacing missing values with "-999". This strategy can be used if there is a belief that the values are not missing at random because it makes the missing values very different from any observed values. However, this method also has its problems, e.g. that it changes the distribution of the data.

As previously mentioned, the random forest model can handle missing data using surrogate splits and is therefore fitted on the data as is. However, we note that the biochemical

variables cholesterol, creatinine and HbA1c are log-transformed in the baseline data set as they were for the Cox model.

2.5 Data Imbalance

Class imbalance is a well studied area in classification (Japkowicz and Stephen, 2002). A variety of sampling methods can be used to address the imbalance, e.g. downsampling, upsampling, and synthetic oversampling techniques (Chawla et al., 2002; Menardi and Torelli, 2014).

We have used downsampling as it is the simplest method. It is used both to balance the classes created by dichotomization (denoted as *Y1*, *Y3* and *Y5* if downsampled based on the dichotomization over year 1, 3 or 5, respectively) but also the censoring (denoted *cens.*), i.e. downsampling the censored observations such that we have an equal number of censored and uncensored observations. This is done because recent methods for dealing with the imbalance by resampling shows that it improved the performance (Afrin et al., 2018). If no downsampling is done it is denoted *none*.

2.6 Model Evaluation

2.6.1 Cross-Validation

The performance is evaluated in a 5 fold cross validation which has been repeated twice (Kohavi et al., 1995). Cross-validation is stratified according to either the censoring or the dichotomized class on which the downsampling has also been done. For the neural network, stratification on the class is also done for the non-downsampled results. All downsampling is done on only the training set and within the cross-validation.

2.6.2 Performance Measures

The performance measures for the survival models have been chosen because they are either routinely reported or account for censoring while they can be calculated for models other than the Cox model (Rahman et al., 2017).

The discriminative powers of the survival models are evaluated using two different

concordance indices (C-indices). The C-index by Harrell Jr et al. (1982) which is calculated as the number of concordant pairs of observations, i.e. pairs where the observation which has the lowest survival time is also predicted to be lowest while the event happened for that observation, over the number of comparable pairs, i.e. pairs where the event happened for the observation with the lowest survival time.

$$C_H = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} I(T_i > T_j)I(\eta_j > \eta_i)\delta_j}{\sum_{i=1}^{n}\sum_{j=1}^{n} I(T_i > T_j)\delta_j} \tag{6}$$

where $\eta_i \in \mathbb{R}$ is a one dimensional score computed for each observation, i.e. the predicted survival time or the linear predictors. T_i is the time to event δ_i is the event status. The C-index by Uno et al. (2011) is given by

$$C_U = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} I(T_i > T_j)I(\eta_j > \eta_i)\delta_j \hat{G}(T_j)^{-2}}{\sum_{i=1}^{n}\sum_{j=1}^{n} I(T_i > T_j)\delta_j \hat{G}(T_j)^{-2}} \tag{7}$$

where $\hat{G(\cdot)}$ is the Kaplan-Meier estimator of the censoring distribution. This index takes censoring into account. Both of the measures lie between 0 and 1 where 0.5 corresponds to random guessing and 1 is the best.

Besides this we report the integrated Brier score (Graf et al., 1999),

$$\text{IBS} = \int_0^{\max(t)} \frac{1}{n}\sum_{i=1}^{n}(I(T_i > t) - \hat{S}(t))^2 dt. \tag{8}$$

for which 0 represents a perfect fit and smaller values are better.

The classification performance is evaluated by the accuracy,

$$\text{accuracy} = \frac{\sum_{i=1}^{n} I(\hat{Y}_i = Y_i)}{n} \tag{9}$$

which measures the proportion of correctly classified observations. It is also evaluated by the sensitivity which measures the proportion of positives correctly identified,

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

where $\text{TP} = I(\hat{Y}_i = 1 \wedge Y_i = 1)$ are the true positives and $\text{FN} = I(\hat{Y}_i = 0 \wedge Y_i = 1)$ are the false negatives. The performance is evaluated by specificity which measures the proportion of negatives correctly identified,

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{11}$$

where $\text{TN} = I(\hat{Y}_i = 0 \wedge Y_i = 0)$ are the true negatives and $\text{FP} = I(\hat{Y}_i = 1 \wedge Y_i = 0)$ are the false positives. Finally, the performance is evaluated by Matthews correlation coefficient (MCC) (Matthews, 1975) because it is a balanced measure that can be used even for highly imbalanced data,

$$\text{MCC} = \frac{\text{TP} \cdot \text{TN} - \text{FP} \cdot \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{12}$$

which gives a value between -1 and 1 where 0 means not better than random and 1 means perfect prediction.

3 Results

The performance of the survival models are improved by including the extracted time model features in terms of C-indices and IBS (Table 3). There is no improvement by using the random forest model even though more data is utilized and it can model interactions.

Downsampling based on censoring gave a small improvement in C_U which takes censoring into account but not C_H and IBS. Downsampling on censoring did not affect the classification performance but this is due to the models already just predicting the majority class (Table 4). When downsampling based on one of the classes (year 1, 3, or 5), the performance in terms of C_H is largely unchanged and in terms of IBS worse, possibly due to the reduced number of observations. In terms of C_U it however improves compared to not having resampled. The classification accuracy of the survival models decreases when the majority class (high survival time) is downsampled. This is because it no longer just predicts a high survival time which can be seen by the specificity increasing. MCC also increases when the majority class is downsampled.

Table 3: Performance of survival methods. Mean and standard deviation. Two methods are compared Cox proportional hazards model (Cox) and the conditional inference forest (CIF) on two data sets baseline (B) and time model (T) (see Section 2.2). The sampling refers to whether the data was downsampled on censoring (cens.), year 1 (Y1), 3 (Y3) or 5 (Y5), or not at all (None).

Method	Data	Sampling	C_H	C_U	IBS
Cox	B	None	0.69 (0.02)	0.64 (0.06)	0.18 (0.01)
Cox	T	None	0.71 (0.02)	0.65 (0.04)	0.18 (0.01)
CIF	B	None	0.68 (0.01)	0.53 (0.10)	0.19 (0.01)
CIF	T	None	0.68 (0.01)	0.54 (0.07)	0.19 (0.01)
Cox	B	cens.	0.69 (0.03)	0.64 (0.05)	0.18 (0.01)
Cox	T	cens.	0.71 (0.04)	0.66 (0.03)	0.19 (0.02)
CIF	B	cens.	0.69 (0.01)	0.58 (0.07)	0.19 (0.01)
CIF	T	cens.	0.69 (0.01)	0.61 (0.05)	0.19 (0.01)
CIF	B	Y1	0.66 (0.01)	0.60 (0.04)	0.26 (0.01)
CIF	T	Y1	0.66 (0.01)	0.60 (0.04)	0.26 (0.01)
CIF	B	Y3	0.68 (0.01)	0.60 (0.04)	0.24 (0.01)
CIF	T	Y3	0.68 (0.02)	0.64 (0.05)	0.25 (0.01)
CIF	B	Y5	0.69 (0.01)	0.64 (0.02)	0.22 (0.01)
CIF	T	Y5	0.69 (0.01)	0.65 (0.03)	0.22 (0.01)

The performance of the neural network is also improved by downsampling. The accuracy increases and the sensitivity and specificity become more balanced. MCC does however only increase for year 5 due to high specificity for the other years before downsampling. For all three years, the neural network performs better than the conditional inference forest.

4 Discussion

The Cox proportional hazards model is the most commonly used model for survival data (Harrell, 2015). The model is easily interpretable due to its simplicity and because it allows for hypothesis testing for relevance of included variables (Cox, 1972). However, due to being a proportional hazards model it cannot give predictions of the actual time-to-event because the base hazard is unknown. It further has the issue that it cannot handle missing data.

The latter two issues are addressed by the conditional inference forest model. It handles missing data by using surrogate splits (Hothorn et al., 2010). Because the methods are have

Table 4: Classification results in percent (except for MCC). Mean and standard deviations. Two methods are compared the conditional inference forest (CIF) and a neural network (NN) on two data sets baseline (B) and time model (T) (see Section 2.2). The sampling refers to whether the data was downsampled on censoring (cens.), year 1 (Y1), 3 (Y3) or 5 (Y5), or not at all (None). Class is the year on which the dichotomized response is cut-off.

Method	Data	Sampling	Class	Accuracy	Sensitivity	Specificity	MCC
CIF	B	None	1	96.2 (0.43)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
CIF	T	None	1	96.2 (0.43)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
NN	B	None	1	95.7 (1.2)	50.2 (20.4)	99.1 (0.3)	0.60 (0.17)
CIF	B	cens.	1	96.2 (0.55)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
CIF	T	cens.	1	96.2 (0.55)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
CIF	B	Y1	1	67.5 (2.7)	72.7 (4.7)	67.3 (2.8)	0.16 (0.02)
CIF	T	Y1	1	70.2 (2.2)	70.5 (6.2)	70.2 (2.4)	0.17 (0.02)
NN	B	Y1	1	81.1 (5.6)	77.0 (17.4)	81.4 (5.4)	0.36 (0.13)
CIF	B	None	3	89.3 (0.99)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
CIF	T	None	3	89.3 (0.99)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
NN	B	None	3	83.6 (3.9)	37.8 (21.3)	97.0 (1.5)	0.42 (0.26)
CIF	B	cens.	3	89.3 (1.1)	0.08 (0.26)	100 (0.00)	0.01 (0.03)
CIF	T	cens.	3	89.3 (1.1)	0.00 (0.00)	100 (0.00)	0.00 (0.00)
CIF	B	Y3	3	66.1 (2.3)	73.0 (3.4)	65.3 (2.7)	0.24 (0.02)
CIF	T	Y3	3	65.8 (2.3)	72.7 (3.8)	65.0 (2.7)	0.24 (0.03)
NN	B	Y3	3	68.4 (10.7)	62.5 (5.8)	70.2 (13.8)	0.30 (0.14)
CIF	B	None	5	80.6 (1.4)	3.74 (1.7)	99.4 (0.47)	0.11 (0.03)
CIF	T	None	5	80.5 (1.5)	1.15 (0.79)	99.9 (0.13)	0.07 (0.04)
NN	B	None	5	68.6 (7.6)	57.4 (7.8)	78.5 (8.5)	0.37 (0.16)
CIF	B	cens.	5	81.0 (1.4)	8.29 (1.8)	98.8 (0.05)	0.18 (0.04)
CIF	T	cens.	5	80.8 (1.5)	3.14 (1.19)	99.8 (0.16)	0.13 (0.04)
CIF	B	Y5	5	65.8 (1.6)	71.7 (3.8)	64.4 (2.4)	0.29 (0.02)
CIF	T	Y5	5	64.9 (1.6)	72.6 (4.2)	63.1 (2.5)	0.27 (0.03)
NN	B	Y5	5	71.0 (2.9)	67.3 (1.9)	74.3 (6.5)	0.42 (0.06)

different strategies for missing values the performance is compared on different subsets of the data. This means that the results are not controlled such that difference in the results are due to one difference in the approach. However, this is the reality of clinical data so a comparison of methods has to consider missing data differently for different methods. The conditional inference forest can further natively give predictions of survival times. These advantages come at the expense of a more complicated model. It is not easily interpretable, though an experimental variable importance measure is available for the conditional inference forest (Hothorn et al., 2018) and a variable importance measure

is available for the random survival forest (Ishwaran et al., 2007). In our data set the conditional inference forest and the Cox model perform similar to each other but the Cox performs slightly better. This could be the case if the assumptions of the Cox model are satisfied or if the flexibility of the random forest is not needed.

That the conditional inference forest model can give a prediction of the time-to-event gives the possibility to compare its performance to methods for the dichotomized response. Generally, dichotomizing is not advisable (Fedorov et al., 2009). However, one might have non-statistical reasons that this form of the response is of interest, e.g. easier translation of knowledge to clinicians. If one chooses this approach, then survival models are no longer a natural choice of model as the problem becomes a binary classification. However, one advantage of using a conditional inference forest for survival data is that it can be used to classify for any cut off.

We consider a neural network for classification. This implementation has the limitation that it cannot handle missing data. But the main limitation of this approach is that it cannot utilize the full information of the time to insulin because this has been encoded as a binary class label. It does have the advantage of optimizing the model to classification and not the survival time.

If the dichotomization is done such that one class is much larger than the other then there is a need for resampling in order to make the classes even. When downsampling is chosen we lose some data. We therefore lose more information on top of the information lost due to dichotomization. However, dealing with the imbalance of the data is necessary to produce generalizable results. In our data, the neural network outperforms the conditional inference forest for the survival data in all three years when the resampling is done. This means if the dichotomized response is of interest we are better off fitting a model directly to this response. If there is no resampling both methods perform poorly. The same downsampling is required in order to get comparable performance from dichotomizing the survival time predictions from the conditional inference forest meaning that using the model for any cut off did produce desirable results (Table 4).

5 Conclusions

In this final section, we give our recommendations and conclusions based on this study.

Imbalance Based on this study downsampling on censoring in survival prediction has a small but positive effect. It might be relevant to investigate in one's data set. Downsampling in classification or dealing with the imbalance in a different way is however important to produce generalizable results.

Interpretation The Cox model performed well in our case. This model is preferable if the interpretation of the model structure is of interest.

Predictions We found that the models trained specifically to one task performed the best for that task, resulting in the conditional inference forest being outperformed in both settings.

Dichotomization If interested in the dichotomized response then it is preferable to build a model on this response directly rather than dichotomizing post training.

References

Afrin, K., G. Illangovan, S. S. Srivatsa, and S. T. Bukkapatnam (2018). Balanced random survival forests for extremely unbalanced, right censored data. *arXiv preprint arXiv:1803.09177*.

Breiman, L. (1996). Bagging predictors. *Machine learning 24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research 16*, 321–357.

Claesson, R., C. Ignell, N. Shaat, and K. Berntorp (2017). Hba1c as a predictor of diabetes after gestational diabetes mellitus. *Primary care diabetes 11*(1), 46–51.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological) 34* (2), 187–220.

Dagliati, A., S. Marini, L. Sacchi, G. Cogni, M. Teliti, V. Tibollo, P. De Cata, L. Chiovato, and R. Bellazzi (2018). Machine learning methods to predict diabetes complications. *Journal of diabetes science and technology 12* (2), 295–302.

Donders, A. R. T., G. J. Van Der Heijden, T. Stijnen, and K. G. Moons (2006). A gentle introduction to imputation of missing values. *Journal of clinical epidemiology 59* (10), 1087–1091.

Doney, A. S., B. Fischer, G. Leese, A. D. Morris, and C. N. Palmer (2004). Cardiovascular risk in type 2 diabetes is associated with variation at the pparg locus: a go-darts study. *Arteriosclerosis, thrombosis, and vascular biology 24* (12), 2403–2407.

Doney, A. S., S. Lee, G. P. Leese, A. D. Morris, and C. N. Palmer (2005). Increased cardiovascular morbidity and mortality in type 2 diabetes is associated with the glutathione s transferase theta–null genotype: A go-darts study. *Circulation 111* (22), 2927–2934.

Eckner, A. (2012). A note on trend and seasonality estimation for unevenly-spaced time series.

Farmer, A. J., L. R. Rodgers, M. Lonergan, B. Shields, M. N. Weedon, L. Donnelly, R. R. Holman, E. R. Pearson, A. T. Hattersley, et al. (2015). Adherence to oral glucose–lowering therapies and associations with 1-year hba1c: A retrospective cohort analysis in a large primary care database. *Diabetes care*, dc151194.

Fedorov, V., F. Mannino, and R. Zhang (2009). Consequences of dichotomization. *Pharmaceutical Statistics: The Journal of Applied Statistics in the Pharmaceutical Industry 8* (1), 50–61.

Ganz, T., J. Wainstein, S. Gilad, R. Limor, M. Boaz, and N. Stern (2017). Serum asymmetric dimethylarginine and arginine levels predict microvascular and macrovascular complications in type 2 diabetes mellitus. *Diabetes/metabolism research and reviews 33* (2), e2836.

Graf, E., C. Schmoor, W. Sauerbrei, and M. Schumacher (1999). Assessment and comparison of prognostic classification schemes for survival data. *Statistics in medicine 18*(17-18), 2529–2545.

Harrell, F. E. (2015). Cox proportional hazards regression model. In *Regression modeling strategies*, pp. 475–519. Springer.

Harrell Jr, F. E. (2018). *rms: Regression Modeling Strategies*. R package version 5.1-2.

Harrell Jr, F. E., R. M. Califf, D. B. Pryor, K. L. Lee, R. A. Rosati, et al. (1982). Evaluating the yield of medical tests. *Jama 247*(18), 2543–2546.

Hippisley-Cox, J. and C. Coupland (2017). Development and validation of qdiabetes-2018 risk prediction algorithm to estimate future risk of type 2 diabetes: cohort study. *bmj 359*, j5019.

Hippisley-Cox, J., C. Coupland, J. Robson, A. Sheikh, and P. Brindle (2009). Predicting risk of type 2 diabetes in england and wales: prospective derivation and validation of qdscore. *Bmj 338*, b880.

Hothorn, T., P. Buehlmann, S. Dudoit, A. Molinaro, and M. Van Der Laan (2006). Survival ensembles. *Biostatistics 7*(3), 355–373.

Hothorn, T., K. Hornik, C. Strobl, and A. Zeileis (2010). Party: A laboratory for recursive partytioning.

Hothorn, T., K. Hornik, C. Strobl, and A. Zeileis (2018). *party: A Laboratory for Recursive Partytioning*. R package version 1.3-1.

Hothorn, T., K. Hornik, and A. Zeileis (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics 15*(3), 651–674.

Hothorn, T., B. Lausen, A. Benner, and M. Radespiel-Tröger (2004). Bagging survival trees. *Statistics in medicine 23*(1), 77–91.

Ishwaran, H. et al. (2007). Variable importance in binary regression trees and forests. *Electronic Journal of Statistics 1*, 519–537.

Ishwaran, H., U. B. Kogalur, E. H. Blackstone, M. S. Lauer, et al. (2008). Random survival forests. *The annals of applied statistics 2*(3), 841–860.

Janiszewski, P. M., I. Janssen, and R. Ross (2007). Does waist circumference predict diabetes and cardiovascular disease beyond commonly evaluated cardiometabolic risk factors? *Diabetes care 30*(12), 3105–3109.

Japkowicz, N. and S. Stephen (2002). The class imbalance problem: A systematic study. *Intelligent data analysis 6*(5), 429–449.

Kohavi, R. et al. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, Volume 14, pp. 1137–1145. Montreal, Canada.

Lagani, V., F. Chiarugi, S. Thomson, J. Fursse, E. Lakasing, R. W. Jones, and I. Tsamardinos (2015). Development and validation of risk assessment models for diabetes-related complications based on the dcct/edic data. *Journal of Diabetes and its Complications 29*(4), 479–487.

Liljestrand, J., A. Havulinna, S. Paju, S. Männistö, V. Salomaa, and P. Pussinen (2015). Missing teeth predict incident cardiovascular events, diabetes, and death. *Journal of dental research 94*(8), 1055–1062.

Lindström, J. and J. Tuomilehto (2003). The diabetes risk score: a practical tool to predict type 2 diabetes risk. *Diabetes care 26*(3), 725–731.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure 405*(2), 442–451.

Menardi, G. and N. Torelli (2014). Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery 28*(1), 92–122.

Mentch, L. and G. Hooker (2017). Formal hypothesis tests for additive structure in random forests. *Journal of Computational and Graphical Statistics 26*(3), 589–597.

Mudelsee, M. (2014). *Climate time series analysis*. Springer.

Nasejje, J. B., H. Mwambi, K. Dheda, and M. Lesosky (2017). A comparison of the conditional inference survival forest model to random survival forests based on a simulation study as well as on two applications with time-to-event data. *BMC medical research methodology 17*(1), 115.

Nielsen, R., L. Donnelly, A. M. Nielsen, K. Zhou, K. Tsirigos, B. K. Ersbøll, L. Clemmensen, E. Pearson, and R. Gupta (2018). Prediction of type 2 diabetes progression by time to insulin using electronic health records. *(draft intended for a diabetes journal)*.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Rahman, M. S., G. Ambler, B. Choodari-Oskooei, and R. Z. Omar (2017). Review and evaluation of performance measures for survival prediction models in external validation settings. *BMC medical research methodology 17*(1), 60.

Ribeiro, M. T., S. Singh, and C. Guestrin (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144. ACM.

Ripley, B. D. and W. N. Venables (2016). *nnet: Feed-Forward Neural Networks and Multinomial Log-Linear Models*. R package version 7.3-12.

Rubin, D. B. (1976). Inference and missing data. *Biometrika 63*(3), 581–592.

Sattar, N., O. Scherbakova, I. Ford, D. S. J. O'Reilly, A. Stanley, E. Forrest, P. W. MacFarlane, C. J. Packard, S. M. Cobbe, and J. Shepherd (2004). Elevated alanine aminotransferase predicts new-onset type 2 diabetes independently of classical risk factors, metabolic syndrome, and c-reactive protein in the west of scotland coronary prevention study. *Diabetes 53*(11), 2855–2860.

Semeraro, F., G. Parrinello, A. Cancarini, L. Pasquini, E. Zarra, A. Cimino, G. Cancarini, U. Valentini, and C. Costagliola (2011). Predicting the risk of diabetic retinopathy in type 2 diabetic patients. *Journal of Diabetes and its Complications 25*(5), 292–297.

Steck, A. K., F. Dong, B. I. Frohnert, K. Waugh, M. Hoffman, J. M. Norris, and M. J. Rewers (2018). Predicting progression to diabetes in islet autoantibody positive children. *Journal of autoimmunity 90*, 59–63.

Sterne, J. A., I. R. White, J. B. Carlin, M. Spratt, P. Royston, M. G. Kenward, A. M. Wood, and J. R. Carpenter (2009). Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj 338*, b2393.

Strobl, C., A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis (2008). Conditional variable importance for random forests. *BMC Bioinformatics 9*(307).

Strobl, C., A.-L. Boulesteix, A. Zeileis, and T. Hothorn (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics 8*(25).

Uno, H., T. Cai, M. J. Pencina, R. B. D'Agostino, and L. Wei (2011). On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in medicine 30*(10), 1105–1117.

Venables, W. N. and B. D. Ripley (2002). *Modern Applied Statistics with S* (Fourth ed.). New York: Springer. ISBN 0-387-95457-0.

Wager, S. and S. Athey (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association 0*(0), 1–15.

Welling, S. H., H. H. Refsgaard, P. B. Brockhoff, and L. H. Clemmensen (2016). Forest floor visualizations of random forests. *arXiv preprint arXiv:1605.09196*.

Xu, P., J. P. Krischer, and Type 1 Diabetes TrialNet Study Group (2016). Prognostic classification factors associated with development of multiple autoantibodies, dysglycemia, and type 1 diabetes - a recursive partitioning analysis. *Diabetes Care*, dc152292.

Zhou, K., L. A. Donnelly, A. D. Morris, P. W. Franks, C. Jennison, C. N. Palmer, and E. R. Pearson (2014). Clinical and genetic determinants of progression of type 2 diabetes: a direct study. *Diabetes care 37*(3), 718–724.

Appendix D

Help files for R-package multiRDPG

Nielsen, A. M., Witten, D. *multiRDPG: Multiple Random Dot Product Graphs, R package version 1.0.1*

Package 'multiRDPG'

December 9, 2018

Version 1.0.1

Type Package

Title Multiple Random Dot Product Graphs

Description Fits the Multiple Random Dot Product Graph Model and performs a test for whether two networks come from the same distribution. Both methods are proposed in Nielsen, A.M., Witten, D., (2018) ``The Multiple Random Dot Product Graph Model", arXiv preprint <arXiv:1811.12172> (Submitted to Journal of Computational and Graphical Statistics).

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Agnes Martine Nielsen [aut, cre], Daniela Witten [aut]

Maintainer Agnes Martine Nielsen <agni@dtu.dk>

Repository CRAN

Date/Publication 2018-12-09 13:40:03 UTC

R topics documented:

multiRDPG *Fitting Multiple Random Dot Product Graphs*

Description

multiRDPG is used to fit Multiple Random Dot Product Graphs from a set of adjacency matrices.

Usage

```
multiRDPG(A, d, maxiter = 100, tol = 1e-06)
```

Arguments

A	List of adjacency matrices representing graphs. Each matrix must be symmetric. All matrices of the same size n x n.
d	Dimension of latent space. d<= n.
maxiter	Maximal number of iterations. Default is 100.
tol	Tolerance for update of the objective function. Default is 1e-6.

Value

Returns a list of the following:

U	Matrix of the joint vectors. n x d.
Lambda	List of diagonal matrices. One for each graph. d x d.
Converged	Represent of the algorithm converged. 1 if converged, 0 if not.
iter	Number of iterations
maxiter	Maximal number of iterations.Default is 100.
objfun	Value of the objective function. sum_k ‖A^k - U Lambda U^T‖_F^2

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG_test

Examples

```
#simulate data
U <- matrix(0, nrow=20, ncol=3)
U[,1] <- 1/sqrt(20)
U[,2] <- rep(c(1,-1), 10)/sqrt(20)
U[,3] <- rep(c(1,1,-1,-1), 5)/sqrt(20)
```

```
L<-list(diag(c(11,6,2)),diag(c(15,4,1)))
A <- list()
for(i in 1:2){
  P <- U%*%L[[i]]%*%t(U)
  A[[i]] <-apply(P,c(1,2),function(x){rbinom(1,1,x)})
  A[[i]][lower.tri(A[[i]])]<-t(A[[i]])[lower.tri(A[[i]])]
}

#fit model
multiRDPG(A,3)
```

`multiRDPG_test` *Performs test based on Multiple Random Dot Product Graph*

Description

`multiRDPG_test` calculates the likelihood ratio test for whether a set of graphs comes from the same disribution.

Usage

```
multiRDPG_test(A, d, maxiter = 100, tol = 1e-06, B = 1000)
```

Arguments

`A`	List of symmetric A matrices
`d`	Dimension of the latent space
`maxiter`	Maximum number of iterations in the fit of multiRDPG. Default is 100.
`tol`	Tolerance for the step in the objective function in multiRDPG. Default is 1e-6.
`B`	Number of permutation iterations. Default is 1000.

Value

Returns a list of the following elements:

`pvalue`	Estimated p-values
`Tval`	Value of the test statistic
`Tstar`	Vector of the test statistic for each permutation iteration
`nullmodel`	Model fit under the null
`altmodel`	Modelfit under the alternative

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG

Examples

```
#simulate data
U <- matrix(0, nrow=20, ncol=3)
U[,1] <- 1/sqrt(20)
U[,2] <- rep(c(1,-1), 10)/sqrt(20)
U[,3] <- rep(c(1,1,-1,-1), 5)/sqrt(20)

L<-list(diag(c(11,6,2)),diag(c(15,4,1)))
A <- list()
for(i in 1:2){
  P <- U%*%L[[i]]%*%t(U)
  A[[i]] <-apply(P,c(1,2),function(x){rbinom(1,1,x)})
  A[[i]][lower.tri(A[[i]])]<-t(A[[i]])[lower.tri(A[[i]])]
}

#perform test
multiRDPG_test(A,3,B=100)
```

`nullestimation` `nullestimation` *calculates the estimation under the null hypothesis*

Description

`nullestimation` calculates the estimation under the null hypothesis

Usage

```
nullestimation(A, d)
```

Arguments

`A`	List of symmetric A matrices
`d`	Dimension of the latent space

Value

Returns a list of the following

`U`	The common latent space vectors. U in R^n x d
`Lambda`	List of Lambdas. Each is a positive diagonal matrix of size d x d.

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG

Examples

```
#simulate data
U <- matrix(0, nrow=20, ncol=3)
U[,1] <- 1/sqrt(20)
U[,2] <- rep(c(1,-1), 10)/sqrt(20)
U[,3] <- rep(c(1,1,-1,-1), 5)/sqrt(20)

L<-list(diag(c(11,6,2)),diag(c(15,4,1)))
A <- list()
for(i in 1:2){
  P <- U%*%L[[i]]%*%t(U)
  A[[i]] <-apply(P,c(1,2),function(x){rbinom(1,1,x)})
  A[[i]][lower.tri(A[[i]])]<-t(A[[i]])[lower.tri(A[[i]])]
}

#fit model
nullestimation(A,3)
```

---

plot.multiRDPGfit *Plots object from* multiRDPG

Description

Plots object from multiRDPG

Usage

```
## S3 method for class 'multiRDPGfit'
plot(x, ...)
```

Arguments

x	multiRDPGfit object from function multiRDPG
...	further arguments passed to or from other methods

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG

---

plot.multiRDPGtest *Plots object from* multiRDPG_test

Description

Plots histogram of permutation test statistics and indicates test statistic value with red line.

Usage

```
## S3 method for class 'multiRDPGtest'
plot(x, ...)
```

Arguments

x	multiRDPGtest object from function multiRDPG_test
...	further arguments passed to or from other methods

Details

Red line indicates the value of the test statistics with a red line.

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG_test

---

print.multiRDPGfit *Print object from* multiRDPG

Description

Print object from multiRDPG

Usage

```
## S3 method for class 'multiRDPGfit'
print(x, ...)
```

Arguments

x	multiRDPGfit object from function multiRDPG
...	further arguments passed to or from other methods

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG

---

print.multiRDPGtest *Print object from* multiRDPG_test

Description

Print object from multiRDPG_test

Usage

```
## S3 method for class 'multiRDPGtest'
print(x, ...)
```

Arguments

x	multiRDPGtest object from function multiRDPG_test
...	further arguments passed to or from other methods

Author(s)

Agnes Martine Nielsen (agni@dtu.dk)

See Also

multiRDPG_test

Index